

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/276921015>

Administració Avançada de GNU/Linux, 4ed 2014

Book · September 2014

CITATIONS

0

READS

83

2 authors:



[Josep Jorba Esteve](#)

Universitat Oberta de Catalunya

70 PUBLICATIONS 241 CITATIONS

[SEE PROFILE](#)



[Remo Suppi](#)

Autonomous University of Barcelona

84 PUBLICATIONS 186 CITATIONS

[SEE PROFILE](#)

Administració avançada del sistema operatiu GNU/Linux

Josep Jorba Esteve
Remo Suppi Boldrito

PID_00212463

Material docent de la UOC

**Josep Jorba Esteve**

Enginyer Superior en Informàtica.
Doctor enginyer en Informàtica per
la UAB. Professor dels Estudis
d'Informàtica, Multimèdia i
Telecomunicacions de la UOC,
Barcelona.

**Remo Suppi Boldrito**

Enginyer de Telecomunicacions.
Doctor en Informàtica per la UAB.
Professor del Departament
d'Arquitectura de Computadors i
Sistemes Operatius per la
Universitat Autònoma de Barcelona.

Quarta edició: setembre 2014

© Josep Jorba Esteve, Remo Suppi Boldrito

© Fundació per a la Universitat Oberta de Catalunya

Av. Tibidabo, 39-43, 08035 Barcelona

Disseny: Manel Andreu

Realització editorial: Oberta UOC Publishing, SL

Dipòsit legal: B-19.881-2014

@ 2014, FUOC. Es garanteix el permís per a copiar, distribuir i modificar aquest document segons els termes de la GNU Free Documentation License, Version 1.2 o qualsevol altra de posterior publicada per la Free Software Foundation, sense seccions invariants ni textos de coberta anterior o posterior. Hi ha una còpia de la llicència en l'apartat "GNU Free Documentation License" d'aquest document.

Introducció

Els sistemes GNU/Linux han arribat a un grau de maduresa important que els fa vàlids per a integrar-los en qualsevol ambient de treball, des de l'escriptori del PC personal fins al servidor d'una gran empresa. L'objectiu principal d'aquest curs és introduir-nos en el món de l'administració dels sistemes GNU/Linux.

Aprendrem com proporcionar, des de GNU/Linux, els serveis necessaris per a diferents ambients d'usuaris i màquines. El camp de l'administració de sistemes és enorme, hi ha moltes tasques, molts problemes que cal tractar, cal tenir grans coneixements de maquinari i programari, i no és sobrer aplicar una mica de psicologia per a tractar amb els usuaris finals dels sistemes.

El curs no pretén abordar una distribució GNU/Linux particular, però se n'han escollit un parell per a tractar els exemples: Debian i Fedora. Respecte al camp de l'administració, s'intentarà gestionar des del nivell més baix possible, normalment l'indicador d'ordres i els fitxers de configuració. Es comentaran, en cada cas, eines de nivell més alt, però cal anar amb compte amb aquestes, ja que solen ser fortament dependents de la distribució utilitzada i fins i tot de la versió; a més, aquestes eines solen variar molt entre versions. L'administració de baix nivell sol ser molt més dura, però sabem on estem actuant, on podem veure els resultats i, a més, ens aporta molts coneixements addicionals sobre les diferents tecnologies utilitzades.

La primera edició d'aquests materials es va fer al final del 2003, i per aquesta raó se n'han anat desenvolupant quatre edicions, que s'han treballat sobre les diferents versions de les distribucions Debian i Fedora disponibles fins al moment actual. D'aquestes distribucions cal destacar que, essencialment, no han canviat en els aspectes bàsics d'administració però sí que han inclòs noves opcions, mòduls, versions del nucli, modes de configuració, etc. La distribució Debian és un paradigma dins del moviment del programari lliure, ja que no pertany a cap empresa i està confeccionat només per les aportacions dels voluntaris distribuïts arreu del món. Debian, a més, integra exclusivament programari lliure (se'n pot afegir d'altre apart). Per altra banda, Fedora és la distribució que té, a més de la seva comunitat, el suport d'una de les empreses més solvents del panorama comercial (Red Hat), i, potser per això, és la que ofereix més suport en l'àmbit empresarial (mitjançant serveis de pagament). A Debian el suport depèn dels voluntaris i del coneixement compartit dels usuaris.

Com que l'administració de sistemes és un camp molt ampli, aquest manual només pretén introduir-nos en aquest apassionant (i com no, també de ve-

gades frustrant) món. Veurem algunes de les tasques típiques i com tractar els diferents problemes, si bé l'administració és un camp que s'aprèn dia a dia, amb l'esforç continuat. I des d'aquí advertim que aquest manual és un treball obert, que amb els seus encerts i els més que probables errors, es pot veure complementat amb els comentaris de seus (patidors) usuaris, de manera que són benvinguts els comentaris i els suggeriments de qualsevol tipus per a la millora dels materials. Aquesta última adaptació es fa sobre una nova estructura del màster adaptat a l'Espai Europeu d'Educació Superior (EEES), que l'adequa a les noves metodologies d'aprenentatge, de conformitat amb la normativa vigent d'àmbit estatal.

Agraïments

Els autors agraeixen a la Fundació per a la Universitat Oberta de Catalunya (<http://www.uoc.edu>) el finançament de l'edició d'aquesta obra, emmarcada en el màster internacional de Programari lliure ofert per aquesta institució.

Continguts

Mòdul didàctic 1

El nucli Linux

Josep Jorba Esteve

1. El nucli del sistema GNU/Linux
2. Personalització o actualització del nucli
3. Procés de configuració i compilació
4. Aplicació de pegats del nucli
5. Mòduls del nucli
6. Virtualització en el nucli
7. Present del nucli i alternatives
8. Taller de configuració del nucli a las necessitats de l'usuari

Mòdul didàctic 2

Administració de servidors

Remo Suppi Boldrito

1. Administració de servidors
 - 1.1. Sistema de noms de domini (*domain name system, DNS*)
 - 1.2. NIS (YP)
 - 1.3. Serveis de connexió remota: telnet i ssh
 - 1.4. Serveis de transferència de fitxers: FTP
 - 1.5. Serveis d'intercanvi d'informació a nivell d'usuari
 - 1.6. Active Directory Domain Controller amb Samba4
 - 1.7. *Internet Message Access Protocol* (IMAP)
 - 1.8. Grups de discussió
 - 1.9. *World Wide Web* (httpd)
 - 1.10. Servidor de WebDav
 - 1.11. Servei de *proxy*: Squid
 - 1.12. OpenLdap (Ldap)
 - 1.13. Serveis d'arxius (NFS, *Network File System*)
 - 1.14. Servidor de wiki
 - 1.15. Gestió de còpies de seguretat (*backups*)
 - 1.16. Public Key Infrastructure -PKI-

Mòdul didàctic 3

Administració de dades

Remo Suppi Boldrito

1. Administració de dades
 - 1.1. PostgreSQL
 - 1.2. El llenguatge SQL
 - 1.3. MySQL
 - 1.4. MariaDB

- 1.5. SQLite
- 1.6. Source Code Control System
- 1.7. Mantis Bug Tracker

Mòdul didàctic 4

Administració de seguretat

Josep Jorba Esteve

- 1. Tipus i mètodes dels atacs
- 2. Seguretat del sistema
- 3. Seguretat local
- 4. SELinux
- 5. Seguretat en xarxa
- 6. Eines de seguretat: detecció de vulnerabilitats i intrusions
- 7. Protecció mitjançant filtratge (*TCP wrappers* i tallafocs)
- 8. Anàlisi de registres
- 9. Taller: anàlisi de la seguretat mitjançant eines

Mòdul didàctic 5

Sintonització, optimització i alta disponibilitat

Remo Suppi Boldrito

- 1. Sintonització, optimització i alta disponibilitat
 - 1.1. Aspectes bàsics
 - 1.2. Monitoratge
 - 1.3. Alta disponibilitat en Linux (High-Availability Linux)

Mòdul didàctic 6

Clúster, Cloud i DevOps

Remo Suppi Boldrito

- 1. Clúster, Cloud i DevOps
 - 1.1. Virtualització
 - 1.2. Beowulf
 - 1.3. Beneficis del còmput distribuït
 - 1.4. Memòria compartida. Models de fils (*threading*)
 - 1.5. OpenMP
 - 1.6. MPI, *Message Passing Interface*
 - 1.7. Rocks Cluster
 - 1.8. FAI
 - 1.9. Logs

GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.
<<http://fsf.org/>>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document “free” in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes

limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Doc-

ument, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody

can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright (C) YEAR YOUR NAME.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with ... Texts.” line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

El nucli Linux

Josep Jorba Esteve

PID_00212470

Índex

Introducció	5
Objectius	6
1. El nucli del sistema GNU/Linux	7
2. Personalització o actualització del nucli	16
3. Procés de configuració i compilació	19
3.1. Compilació de les branques 2.6.x i 3.x del kernel Linux	21
3.2. Compilació del nucli en Debian (<i>Debian Way</i>)	29
4. Aplicació de pedaços (<i>patch</i>) al nucli	34
5. Mòduls del nucli	37
5.1. DKMS: mòduls recompilats dinàmicament	40
6. Virtualització en el nucli	43
6.1. KVM	45
6.2. VirtualBox	51
6.3. Xen	55
7. Present del nucli i alternatives	62
8. Taller de configuració del nucli a les necessitats de l'usuari	67
8.1. Configuració del nucli en Debian	67
8.2. Configuració del nucli en Fedora/Red Hat	69
8.3. Configuració d'un nucli genèric	71
Resum	74
Activitats	75
Bibliografia	75

Introducció

El nucli (en anglès *kernel*) del sistema operatiu GNU/Linux (que habitualment anomenem simplement Linux) [Vasb] és la part central del sistema: s'encarrega de posar-lo en funcionament i, una vegada aquest ja és utilitzable per les aplicacions i els usuaris, s'encarrega de gestionar els recursos de la màquina, controlant la gestió de la memòria, els sistemes de fitxers, les operacions d'entrada i sortida, els processos i la seva intercomunicació.

L'origen es remunta a l'any 1991, quan a l'agost, un estudiant finlandès anomenat **Linus Torvalds** va anunciar en una llista de notícies que havia creat el seu propi nucli de sistema operatiu, que funcionava conjuntament amb programari GNU, i l'oferia a la comunitat de desenvolupadors perquè el provés i suggerís millores per fer-lo més utilitzable. Aquest és l'origen del nucli del sistema operatiu que més tard s'anomenaria **GNU/Linux**.

Una de les particularitats de Linux és que, seguint la filosofia de programari lliure, se'ns ofereix el codi font del nucli del sistema operatiu (del *kernel*), de manera que és una eina perfecta per a l'educació en temes d'anàlisi i disseny de sistemes operatius.

L'altre avantatge principal és que, com que disposem dels arxius de codi font, podem compilar-los per a adaptar-los millor al nostre sistema i, com veurem en el mòdul "Sintonització, optimització i alta disponibilitat", configurar-los posteriorment per a donar un millor rendiment al sistema.

En aquest mòdul veurem com gestionar aquest procés de preparació d'un nucli per al nostre sistema GNU/Linux: com, partint dels arxius font, podem obtenir una nova versió del nucli adaptada al nostre sistema. Veurem com es desenvolupen les fases de configuració, es compila posteriorment i es fan proves amb el nou nucli obtingut.

A més, examinarem com el nucli ha anat afegint tota una sèrie de característiques al llarg de la seva evolució, que l'han convertit en competitiu davant d'altres sistemes. En especial, observarem algunes característiques de la virtualització que se'ns ofereixen amb suport des del mateix nucli.

Origen de Linux

El nucli Linux es remunta a l'any 1991, quan Linus Torvalds el va oferir per a l'ús de la comunitat. És un dels pocs sistemes operatius àmpliament usats del qual es disposa el codi font.

Objectius

En aquest mòdul es mostren els continguts i les eines procedimentals per aconseguir els objectius següents:

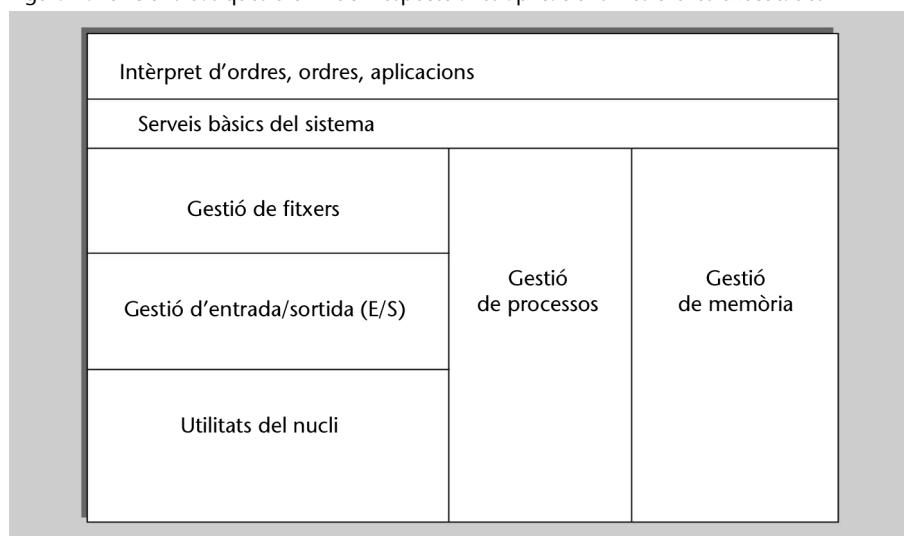
- 1.** Conèixer el funcionament del nucli (*kernel*) i dels processos de configuració associats.
- 2.** Poder configurar i crear el nucli del sistema en les distribucions més habituals.
- 3.** Entendre l'ús dels mòduls del nucli i decidir-ne la integració (o no) dins de la part estàtica del nucli.
- 4.** Conèixer les tècniques de virtualització i, en particular, de les incloses en el nucli.
- 5.** Saber adaptar el nucli a les necessitats particulars de l'usuari, per a sintetitzar els seus sistemes.

1. El nucli del sistema GNU/Linux

El nucli o *kernel* és la part bàsica de qualsevol sistema operatiu [Tan87, Tan06], sobre la qual descansa el codi dels serveis fonamentals per a controlar el sistema complet. Bàsicament, la seva estructura pot separar-se en una sèrie de components o de mòduls de gestió orientats al següent:

- **Gestió de processos:** quines tasques s'executaran i en quin ordre i amb quina prioritat. Un aspecte important és la planificació de la CPU: com s'optimitza el temps de la CPU per a executar les tasques amb el màxim rendiment o interactivitat possibles amb els usuaris?
- **Intercomunicació de processos i sincronització:** com es comuniquen tasques entre elles, amb quins diferents mecanismes i com poden sincronitzar-se grups de tasques?
- **Gestió d'entrada/sortida (E/S):** control de perifèrics i gestió de recursos associats.
- **Gestió de memòria:** optimització de l'ús de la memòria, sistema de paginació i memòria virtual.
- **Gestió de fitxers:** com el sistema controla i organitza els fitxers presents en el sistema, i com hi accedeix.

Figura 1. Funcions bàsiques d'un nucli respecte a les aplicacions i les ordres executades



En els sistemes privatis, el nucli, o *kernel*, està perfectament “ocult” sota les capes del programari del sistema operatiu, i l’usuari final no té una perspectiva clara de què és aquest nucli ni té tampoc cap possibilitat de canviar-lo o optimitzar-lo, si no és per l’ús d’esotèrics editors de “registres” interns, o programes especialitzats de tercers (normalment d’alt cost). A més, el nucli sol ser únic, és el que proporciona el fabricant, el qual es reserva el dret d’introduir-hi les modificacions que vulgui i quan vulgui, i tractar els errors que apareguin en terminis no estipulats, mitjançant actualitzacions que ens ofereix com a “pedaços” d’errors (o grups de pedaços anomenats comunament *service packs* o *updates*).

Un dels principals problemes d’aquesta aproximació és precisament la disponibilitat d’aquests pedaços: disposar de les actualitzacions dels errors a temps i, si es tracta de problemes de seguretat, encara amb més raó, ja que fins que no estiguin corregits no podem garantir la seguretat del sistema per a problemes ja coneguts. Moltes organitzacions, grans empreses, governs, institucions científiques i militars no poden dependre dels capricis d’un fabricant per a solucionar els problemes de les seves aplicacions crítiques.

En aquest cas, el nucli Linux ofereix una solució de codi obert, amb els conseqüents permisos de modificació, correcció i possibilitat que qualsevol que vulgui i tingui els coneixements adequats en generi noves versions i actualitzacions de manera ràpida. Això permet als usuaris amb necessitats crítiques controlar-ne millor les aplicacions i el mateix sistema, i poder elaborar sistemes amb el sistema operatiu “a la carta”, personalitzat al gust de cada usuari final. També permet disposar, al seu torn, d’un sistema operatiu amb codi obert, desenvolupat per una comunitat de programadors coordinats mitjançant Internet i accessible, per a educació, per a disposar del codi font i abundant documentació, o per a la producció final dels sistemes GNU/Linux adaptats a necessitats individuals o d’un determinat col·lectiu.

En disposar del codi font, es poden aplicar millores i solucions de manera immediata, a diferència del programari privat, on hem d’esperar les actualitzacions del fabricant. Podem, a més, personalitzar el nucli tant com necessitem, requisit essencial, per exemple, en aplicacions d’alt rendiment, crítiques en el temps o en solucions amb sistemes encastats (per a dispositius mòbils o altra electrònica de consum).

A continuació repassem una mica la història del nucli [Kera, Kerb]. El nucli Linux el va començar a desenvolupar un estudiant finès anomenat Linus Torvalds, el 1991, amb la intenció de fer una versió semblant a MINIX [Tan87][Tan06] (versió per a PC d’UNIX [Bac86]) per al processador 386 d’Intel. La primera versió publicada oficialment va ser la de Linux 1.0 el març de 1994, en la qual s’inclouia només l’execució per a l’arquitectura i386 i suportava màquines d’un sol processador. Linux 1.2 va ser publicat el març de 1995 i va ser la primera versió que donava cobertura a diferents arquitectures, com

MINIX

El nucli té l’origen en el sistema MINIX, desenvolupat per Andrew Tanenbaum, com a clon d’UNIX per a PC.

ara Alpha, Sparc i Mips. Linux 2.0, el juny de 1996, va afegir més architectures i va ser la primera versió a incorporar suport multiprocessador (SMP) [Tum]. Amb Linux 2.2, del gener de 1999, es van incrementar les prestacions de suport SMP de manera significativa, i es van afegir controladors per a una gran quantitat de maquinari. En la 2.4, el gener de 2001, es va millorar el suport SMP, es van incorporar noves architectures i es van integrar controladors per a dispositius USB, PC Card (PCMCIA dels portàtils), part de PnP (*plug and play*), suport de RAID i volums, etc. En la branca 2.6 del nucli (desembre de 2003), es va millorar sensiblement el suport SMP, es va introduir una millor resposta del sistema de planificació de CPU, l'ús de fils (*threads*) en el nucli, millor suport d'architectures de 64 bits, suport de virtualització i una millor adaptació a dispositius mòbils.

El juliol de 2011, Linus Torvalds va anunciar la versió Linux 3.0, no pels seus grans canvis tecnològics, sinó per iniciar una sèrie de canvis posteriors i per commemorar el 20è aniversari de Linux. Algunes fites més en la branca 3.x:

- 3.3 (març de 2012): es fa una mescla de les fonts de Linux i s'hi incorpora Android, que bàsicament és un kernel Linux amb certes modificacions; després de diversos intents en aquesta versió, es fa la unió de tots dos. S'hi introdueix també la capacitat de poder arrencar directament mitjançant les noves EFI, substitució de les BIOS.
- 3.6 (setembre de 2012): millores en el suport del sistema de fitxers Btrfs (quotes, *snapshots*), pensat per a substituir en el futur a ext4.
- 3.7 (desembre de 2012): s'hi incorpora el suport per a diverses architectures ARM.
- 3.8 (febrer de 2013): com a curiositat, s'elimina el suport a processadors 286, i es fan importants millores en diversos sistemes de fitxers.
- 3.13 (gener de 2014): suport d'NFtables, la següent generació de regles de tallafoc (substituint iptables).
- 3.15 (juny de 2014): es millora el suport d'EFI, l'escriptura en sistemes de fitxers FUSE, i el suport per a repertoris vectorials de noves CPU Intel.

Respecte al procés de desenvolupament, des de la seva creació per Linus Torvalds el 1991 (versió 0.01), ell mateix ha continuat mantenint el nucli, però a mesura que el seu treball li ho permetia i a mesura que el nucli madurava (i creixia), es va veure obligat a mantenir les diferents versions estables del nucli gràcies a diferents col·laboradors, mentre que Linus continua (en la mesura del possible) desenvolupant i recopilant aportacions per a l'última versió de desenvolupament del nucli. Els col·laboradors principals en aquestes versions han estat [Lkm]:

- 2.0 David Weinehall.
- 2.2 Alan Cox (també desenvolupa i publica pedaços per a la majoria de versions).
- 2.4 Marcelo Tosatti.
- 2.5 Linus Torvalds (branca de desenvolupament).
- 2.6 Greg Kroah-Hartman (versions estables, entre altres) / Linus Torvalds, Andrew Morton (*releases* de desenvolupament).
- 3.x Greg Kroah-Hartman / Linus Torvalds (*releases* de desenvolupament).

Per a veure una mica la complexitat del nucli de Linux, vegem una taula amb la seva història resumida en les diferents versions i en la mida respectiva del codi font. A la taula només s'indiquen les versions de producció; la mida està especificada en milers de línies del codi de la font del nucli:

Versió	Data de publicació	Línies de codi (en milers)
0.01	09-1991	10
1.0	03-1994	176
1.2	03-1995	311
2.0	06-1996	649
2.2	01-1999	1.800
2.4	01-2001	3.378
2.6	12-2003	5.930
3.10	06-2013	15.803

Complexitat del nucli

El nucli avui en dia ha assolit uns graus de maduresa i complexitat significatius.

Com podem comprovar, s'ha evolucionat d'unes deu mil línies a sis milions en les primeres versions de la branca 2.6; les últimes versions de la branca 3.x tenen ja més de quinze milions de línies.

En aquests moments, el desenvolupament continua en la branca 3.x del nucli, l'última versió estable, que inclou la majoria de distribucions com a versió principal (algunes encara inclouen 2.6.x, però 3.x acostuma a ser l'opció per defecte en la instal·lació).

Tot i que ara la branca principal és la 3.x, un cert coneixement de les versions anteriors és imprescindible, ja que fàcilment podem trobar màquines amb distribucions antigues que no s'hagin actualitzat i que és possible que hàgim de mantenir o en les quals hàgim de fer un procés de migració a versions més actuals.

En la branca del 2.6, durant el seu desenvolupament es van accelerar de manera significativa els treballs del nucli, ja que tant Linus Torvalds com Andrew Morton (que van mantenir algunes de les branques de Linux 2.6 en desenvolupament) es van incorporar (durant el 2003) a l'Open Source Development Laboratory (OSDL), un consorci d'empreses la finalitat del qual és promociónar l'ús d'Open Source i GNU/Linux en l'empresa (en el consorci es troben, entre moltes altres empreses amb interessos en GNU/Linux, HP, IBM, Sun, Intel, Fujitsu, Hitachi, Toshiba, Red Hat, Suse, Transmeta, etc.). En aquell moment es va donar una situació interessant, ja que el consorci OSDL va fer de

patrocinador dels treballs, tant per al mantenidor de la versió estable del nucli (Andrew) com per al de la de desenvolupament (Linus), treballant a temps complet en les versions i en els temes relacionats. Linus es manté independent, treballant en el nucli, mentre que Andrew va anar a treballar a Google, on continua a temps complet els seus desenvolupaments, fent pedaços amb diferents i noves aportacions al nucli, en el que es coneix com a branca de desenvolupament *-mm*. Després d'un cert temps, OSDL es va reconvertir en la fundació The Linux Foundation.

Cal tenir en compte que amb les versions actuals del nucli s'ha assolit ja un alt grau de desenvolupament i maduresa, cosa que farà que cada vegada s'ampliï més el temps entre la publicació de les versions estables, però no de les revisions parcials o de desenvolupament, aspecte en el qual els mantenidors esperen una nova versió cada dos o tres mesos.

Enllaç d'interès

Linux Foundation:
<http://www.linuxfoundation.org>

A més, un altre factor a considerar és la mida i el nombre de persones que estan treballant en el desenvolupament actual. Al començament hi havia unes quantes persones que tenien un coneixement global de tot el nucli, mentre que avui en dia tenim un gran nombre de persones que el desenvolupen (es creu que prop de diversos milers) amb diferents contribucions, tot i que el grup dur s'estima en unes quantes dotzenes de desenvolupadors.

També es pot tenir en compte que la majoria de desenvolupadors (dels milers) només tenen uns coneixements parcials del nucli i ni tots treballen simultàniament, ni la seva aportació és igual de rellevant (algunes aportacions només corregeixen errors senzills). En l'altre extrem, són unes quantes persones (com els mantenidors) les que disposen d'un coneixement total del nucli. Això implica que es puguin allargar els desenvolupaments i que s'hagin de depurar les aportacions, per a comprovar que no entrin en conflicte entre elles, o que s'hagi d'escollir entre possibles alternatives de prestacions per a un mateix component.

Respecte a la numeració de les versions del nucli de Linux [Ker, Ces06, Lov10], cal tenir en compte els aspectes següents:

1) Fins a la branca del nucli 2.6.x, les versions del nucli Linux es regien per una divisió en dues sèries: una era l'anomenada *experimental* (amb numeració senar en la segona xifra, com 1.3.xx, 2.1.x o 2.5.x) i l'altra era la de producció (sèrie parella, com 1.2.xx, 2.0.xx, 2.2.x, 2.4.x i més). La sèrie experimental eren versions que es movien ràpidament i s'utilitzava per a provar noves prestacions, algorismes, controladors de dispositiu, etc. Per la pròpia naturalesa dels nuclis experimentals, podien tenir comportaments impredecibles, com ara pèrdues de dades, bloquejos aleatoris de la màquina, etc. Per tant, no estaven destinades a utilitzar-se en màquines per a la producció, tret que es volgués provar una característica determinada (amb els consegüents perills).

Els nuclis de producció (sèrie parella) o estables eren els nuclis amb un conjunt de prestacions ben definit, amb un nombre baix d'errors coneguts i controladors de dispositius provats. Es publicaven amb menys freqüència que els experimentals i n'hi havia diverses versions, unes de més o menys qualitat que d'altres. Les distribucions GNU/Linux se solen basar en una determinada versió del nucli estable, no necessàriament l'últim nucli de producció publicat.

2) En la numeració del nucli Linux (utilitzada en la branca 2.6.x), es continuen conservant alguns aspectes bàsics: la versió s'indica per uns números X.Y.Z, en què normalment X és la versió principal, que representa els canvis importants del nucli; Y és la versió secundària, i habitualment implica millores en les prestacions del nucli: Y és parell en els nuclis estables i senar en els desenvolupaments o proves; Z és la versió de construcció, que indica el número de la revisió de X.Y, quant a pedaços o correccions fetes.

En els últims esquemes s'arriba a introduir quarts nombres, per a especificar Z canvis menors, o diferents possibilitats de la revisió (amb diversos pedaços afegits que corregeixen errades). La versió així definida amb quatre nombres és la que es considera estable (*stable*). També es fan servir altres esquemes per a les diverses versions de prova (normalment no recomanables per a entorns de producció), com sufixos `-rc` (*release candidate*) i `-mm`, que són nuclis experimentals amb gran introducció de pedaços que suposen noves prestacions addicionals com proves de diverses tècniques noves; o els `-git`, que són una mena de “foto” diària del desenvolupament del nucli. Aquests esquemes de numeració estan en constant canvi per a adaptar-se a la forma de treballar de la comunitat del nucli i a les seves necessitats per a accelerar el desenvolupament.

Els distribuïdors no acostumen a incloure l'última versió del nucli, sinó aquella que ells han provat amb més freqüència i poden verificar que és estable per al programari i components que contenen. Partint d'aquest esquema de numeració clàssic (que es va seguir durant les branques 2.4.x fins als començaments de la 2.6), hi va haver algunes modificacions per a adaptar-se al fet que el nucli (branca 2.6.x) esdevé més estable (fixant X.Y a 2.6) i cada vegada les revisions són menors (perquè signifiquen un salt de versió dels primers nombres), però el desenvolupament continu i frenètic segueix.

3) En la branca 3.x, quan després del 2.6.39 Linus va decidir renombrar 3.x, el segon nombre s'ha fet servir com a nombre de revisió per seqüència temporal a mesura que sortien els nous kernels, i s'hi pot afegir un tercer nombre per a designar correccions de fallades respecte a *bugs* o seguretat. S'espera que, de manera similar, en un determinat moment es progressi a les branques 4.x, 5.x, etc. Pel que fa a les versions diferents del kernel, ara s'acostuma a denominar *mainline* la versió de desenvolupament que normalment manté Linus, i que apareix de manera periòdica cada dos o tres mesos. Quan apareix alliberada la versió passa a l'estat de *stable*, i s'hi fan diverses iteracions (indicades pel tercer nombre) per *bugs*, normalment de dues a tres per mes, tot i que només hi ha unes poques revisions de l'estable, perquè mentrestant el *mainline*

següent esdevé estable. Com a cas excepcional, hi ha alguns kernels denominats *longterm*, per a propòsits de manteniment de llarga durada, en els quals s'inclou el suport de pedaços que es facin en altres versions, per a portar-los a aquestes. Poden ser, per exemple, kernels que hagin estat escollits per distribucions de tipus LTS (*long term support*) o interessants perquè contenen alguna prestació especial. S'ha de diferenciar que els kernels de distribució acostumen a ser mantinguts per les mateixes distribucions, normalment com a paquets als seus repositoris, i poden incloure nivells de pedaç addicionals propis de la distribució. Aquests kernels especials de les distribucions es poden detectar amb l'ordre `uname -r`, que ens dóna la versió del kernel actual, pels números addicionals als predefinits en la numeració del kernel, inclosos en la sortida de l'ordre.

4) Per a obtenir l'últim nucli publicat (que normalment s'anomena *vanilla* o *pristine*), cal anar a l'arxiu de nuclis Linux (disponible a <http://www.kernel.org>). També podran trobar-s'hi alguns pedaços del nucli original, que corregeixen errors detectats després de la publicació del nucli.

Enllaç d'interès

Dipòsit de nuclis Linux:
<http://www.kernel.org>

Algunes de les característiques tècniques [Ces06, Kan, Lov10] del nucli Linux que podríem destacar són:

- Nucli de tipus monolític: bàsicament és un gran programa creat com una unitat, però conceptualment dividit en diversos components lògics.
- Té suport per a càrrega i descàrrega de porcions del nucli a demanda; aquestes porcions s'anomenen *mòduls* i solen ser característiques del nucli o controladors de dispositiu.
- Fils de nucli: per al funcionament intern s'utilitzen diversos fils (*threads* en anglès) d'execució interns al nucli, que poden estar associats a un programa d'usuari o bé a una funcionalitat interna del nucli. En Linux no es feia un ús intensiu d'aquest concepte originalment, però ha passat a ser un concepte fonamental per al rendiment, en especial a causa de l'aparició de les CPU *multicore*. En les diferents revisions de la branca 2.6.x es va oferir un suport millor, i gran part del nucli s'executa usant diversos fils d'execució.
- Suport d'aplicacions multifil: suport d'aplicacions d'usuari de tipus multifil (*multithread*), ja que molts paradigmes de computació de tipus client/servidor necessiten servidors capaços d'atendre múltiples peticions simultànies dedicant un fil d'execució a cada petició o grup de peticions. Linux té una biblioteca pròpia de fils que pot usar-se per a les aplicacions multifil, amb les millores que es van introduir en el nucli, que també han permès un ús millor per a implementar biblioteques de fils per al desenvolupament d'aplicacions.
- El nucli és de tipus no preemptiu (*nonpreemptive*): això implica que dins del nucli no poden passar-se crides a sistema (en mode supervisor) mentre

s'està resolent la tasca de sistema, i quan aquesta acaba, es prossegueix l'execució de la tasca anterior. Per tant, el nucli dins d'una crida no pot ser interromput per a atendre una altra tasca. Normalment, els nuclis preemptius estan associats a sistemes que treballen en temps real, en què s'ha de permetre la situació anterior per a tractar esdeveniments crítics. Hi ha algunes versions especials del nucli de Linux per a temps real (branques `-rt`, de *realtime*), que ho permeten per mitjà de la introducció d'uns punts fixos en què les tasques del nucli poden interrompre's entre elles. També s'ha millorat especialment aquest concepte en la branca 2.6.x del nucli, que en alguns casos permet interrompre algunes tasques del nucli, reassignables, per a tractar-ne d'altres i prosseguir-ne posteriorment l'execució. Aquest concepte de nucli preemptiu també pot ser útil per a millorar tasques interactives, ja que si es produeixen crides costoses al sistema, poden provocar retards en les aplicacions interactives.

- Suport per a multiprocessador, tant el que s'anomena *multiprocessament simètric* (SMP) com el *multicore*. Aquest concepte sol englobar màquines que van des del cas simple de 2 fins 64 CPU col·locades en diferents sòcols físics de la màquina. Aquest tema s'ha posat d'especial actualitat amb les arquitectures de tipus *multicore*, que permeten de 2 a 8 o més nuclis de CPU en un mateix sòcol físic, en màquines accessibles als usuaris domèstics. Linux pot usar múltiples processadors, i cada processador pot manejar una o més tasques. Però originalment hi havia algunes parts del nucli que disminuïen el rendiment, ja que estan pensades per a una única CPU i obliguen a parar el sistema sencer en determinats bloquejos. SMP és una de les tècniques més estudiades en la comunitat del nucli de Linux, i s'han obtingut millores importants en les branques 2.6 i 3. Del rendiment SMP i *multicore* depèn en gran mesura l'adopció de Linux en els sistemes empresarials, en l'aspecte de sistema operatiu per a servidors.
- Sistemes de fitxers: el nucli té una bona arquitectura dels sistemes de fitxers, ja que el treball intern es basa en una abstracció d'un sistema virtual (VFS, *virtual file system*), que pot ser adaptada fàcilment a qualsevol sistema real. Com a resultat, Linux és potser el sistema operatiu que més sistemes de fitxers suporta, des del seu propi ext2 inicial, fins a msdos, vfat, ntfs, sistemes amb *journal* com ext3, ext4, ReiserFS, JFS(IBM), XFS(Silicon), ZFS (Oracle), Btrfs, NTFS, iso9660 (CD), udf, etc. i s'hi van afegint més en les diferents revisions del nucli.

Altres característiques menys tècniques (una mica de màrqueting) que podríem destacar són:

1) Linux és gratuït: juntament amb el programari GNU, i l'inclòs en qualsevol distribució, podem tenir un sistema tipus UNIX complet pràcticament pel cost del maquinari; i per la part dels costos de la distribució GNU/Linux, podem obtenir-la pràcticament gratis. Però és bo pagar per una distribució completa, amb els manuals i el suport tècnic, a un cost més baix comparat amb el que es paga per alguns sistemes privats, o contribuir amb la seva compra

al desenvolupament de les distribucions que més ens agradin o ens resultin pràctiques.

- 2) Linux és personalitzable: la llicència GPL ens permet llegir i modificar el codi font del nucli (sempre que tinguem els coneixements adequats).
- 3) Linux s'executa en maquinari antic bastant limitat; és possible, per exemple, crear un servidor de xarxa amb un 386 amb 4 MB de RAM (hi ha distribucions especialitzades en recursos baixos i en processadors o arquitectures obsoletes).
- 4) Linux és un sistema d'altres prestacions: l'objectiu principal en Linux és l'eficiència i s'intenta aprofitar al màxim el maquinari disponible.
- 5) Alta qualitat: els sistemes GNU/Linux són molt estables, amb una baixa proporció d'errors, i redueixen el temps dedicat a mantenir els sistemes.
- 6) El nucli és bastant reduït i compacte: és possible col·locar-lo, juntament amb alguns programes fonamentals, en un sol format antic de disc d'1,44 MB (hi ha diverses distribucions d'un sol disquet amb programes bàsics).
- 7) Linux és compatible amb una gran part dels sistemes operatius, pot llegir fitxers de pràcticament qualsevol sistema de fitxers i pot comunicar-se per xarxa per a oferir i rebre serveis de qualsevol d'aquests sistemes. A més, també amb certes biblioteques pot executar programes d'altres sistemes (com MS-DOS, Windows, BSD, Xenix, etc.) en l'arquitectura x86 32 o 64 bits o bé virtualitzar màquines completes, es pot disposar d'imatges virtuals de distribucions GNU/Linux ja preparades per a actuar de sistemes convidats en gestors de virtualització.
- 8) Linux disposa d'un completíssim suport: no hi ha cap altre sistema que tingui la rapidesa i la quantitat de pedaços i actualitzacions que té Linux, ni tan sols en els sistemes privatis. Per a un problema determinat, hi ha infinitat de llistes de correu i fòrums que en poques hores poden permetre solucionar qualsevol problema. L'únic problema és en els controladors de maquinari recent, que molts fabricants encara es resisteixen a proporcionar, si no és per a sistemes privatis. Però això està canviant a poc a poc, i alguns dels fabricants més importants de sectors com els de les targetes de vídeo (NVIDIA, AMD ATI) i impressores (Epson, HP) ja comencen a proporcionar els controladors per als seus dispositius, de codi obert o bé binaris utilitzables pel nucli.

2. Personalització o actualització del nucli

Com a usuaris o administradors de sistemes GNU/Linux, hem de tenir en compte les possibilitats que ens ofereix el nucli per a adaptar-lo a les nostres necessitats i als nostres equips.

Normalment, construïm els nostres sistemes GNU/Linux a partir de la instal·lació en els nostres equips d'alguna de les distribucions de GNU/Linux, tant si són comercials, com Red Hat o Suse, com “comunitàries”, com Debian i Fedora.

Aquestes distribucions aporten, en el moment de la instal·lació, una sèrie de nuclis Linux binaris ja preconfigurats i compilats, i normalment hem d'escollir quin nucli del conjunt dels disponibles s'adapta més bé al nostre maquinari. Hi ha nuclis genèrics per a una arquitectura, per a un model de processador o bé orientats a disposar d'una sèrie de recursos de memòria; d'altres ofereixen una barreja de controladors de dispositius [Cor05], possibilitats de virtualització, etc.

Una altra opció d'instal·lació acostuma a ser la versió del nucli. Normalment les distribucions usen una versió per a la instal·lació que consideren prou estable i provada perquè no causi problemes als usuaris. Per exemple, avui dia moltes distribucions vénen amb una versió de la branca 3.x del nucli per defecte, que es considerava la versió més estable en el moment en què va sortir la distribució. En alguns casos, en el moment de la instal·lació pot oferir-se la possibilitat d'usar com a alternativa versions més modernes, amb un suport millor per a dispositius més moderns (d'última generació), però potser no tan provades.

A més, els distribuïdors acostumen a modificar el nucli per millorar el comportament de la seva distribució o corregir errors que han detectat en el nucli en el moment de les proves. Una altra tècnica bastant comuna en les distribucions comercials és deshabilitar prestacions problemàtiques, que poden causar errors o que necessiten una configuració específica de la màquina. També pot passar que es consideri que una determinada prestació no es prou estable per a incloure-la activada.

Això ens duu a considerar que, per molt bé que un distribuïdor faci la feina d'adaptar el nucli a la seva distribució, sempre ens podem trobar amb una sèrie de problemes o objectius que no podem dur a terme amb la situació actual:

- El nucli no està actualitzat en l'última versió estable disponible; no es disposa de suport per a alguns dispositius moderns.

Personalització del nucli

La possibilitat d'actualitzar i personalitzar el nucli a mida ofereix una bona adaptació a qualsevol sistema, fet que permet l'optimització i la sintonització del nucli al sistema destí.

- El nucli estàndard no disposa de suport per als dispositius que tenim, perquè no han estat habilitats.
- Els controladors que ens ofereix un fabricant necessiten una nova versió del nucli o modificacions.
- A la inversa, el nucli és massa modern i tenim maquinari antic que ja no té suport en els últims nuclis.
- El nucli, tal com està, no obté les màximes prestacions dels nostres dispositius.
- Algunes aplicacions que volem fer servir requereixen suport d'un nucli nou o d'algunes de les seves prestacions.
- Volem estar a l'última i ens arrisquem instal·lant últimes versions del nucli Linux.
- Ens agrada investigar o provar els nous avenços del nucli o bé volem tocar-lo o modificar-lo.
- Volem programar un controlador per a un dispositiu no suportat.
- Etc.

Per aquests i altres motius podem no estar contents amb el nucli que tenim. Se'ns plantegen llavors dues possibilitats: actualitzar el nucli binari de la distribució o bé personalitzar-lo a partir dels paquets font.

Veurem a continuació algunes qüestions relacionades amb les diferents opcions i què impliquen:

1) Actualització del nucli de la distribució. El distribuïdor normalment publica també les actualitzacions del nucli que van apareixent. Quan la comunitat Linux crea una nova versió del nucli, cada distribuïdor la uneix a la seva distribució i en fa les proves pertinents. Després del període de prova, s'identifiquen possibles errors, els corregeix i produeix l'actualització del nucli pertinent respecte al que oferia en els CD de la distribució. Els usuaris poden descarregar la nova revisió de la distribució del lloc web o bé actualitzar-la mitjançant algun sistema automàtic de paquets via dipòsit. Normalment, es verifica quina versió té el sistema, es descarrega el nucli nou i es fan els canvis necessaris perquè la propera vegada el sistema funcioni amb el nucli nou, i es manté la versió antiga per si hi ha problemes.

Aquesta mena d'actualització ens simplifica molt el procés, però no necessàriament ha de solucionar els nostres problemes, ja que pot ser que el nostre maquinari no estigui encara suportat o que la característica que volem provar del nucli no estigui encara en la versió que tenim de la distribució; cal recordar que no ha de fer servir per força l'última versió disponible (per exemple a kernel.org), sinó aquella que el distribuïdor consideri estable per a la seva distribució.

Si el nostre maquinari tampoc no ve habilitat per defecte en la nova versió, ens trobem en la mateixa situació. O senzillament, si volem la darrera versió, aquest procés no ens serveix.

2) Personalització del nucli. En aquest cas, anirem als paquets font del nucli i adaptarem “a mà” el maquinari o les característiques que volem. Passarem per un procés de configuració i compilació dels paquets font del nucli per a, finalment, crear un nucli binari que instal·larem en el sistema i, així, el tindrem disponible en la següent arrencada del sistema.

També aquí podem trobar-nos amb dues opcions més: o bé per defecte obtenim la versió “oficial” del nucli (kernel.org) o bé podem acudir als paquets font proporcionats per la mateixa distribució. Cal tenir en compte que distribucions com Debian i Fedora fan una tasca important d’adequació del nucli i de correcció d’errors que afecten la seva distribució, i gràcies a això podem disposar, en alguns casos, de correccions addicionals al codi original del nucli. Una vegada més, els paquets font oferts per la distribució no necessàriament han de correspondre a l’última versió estable publicada.

Vegeu també

La personalització del nucli és un procés que es descriu amb detall en els apartats següents.

Aquest sistema ens permet la màxima fiabilitat i control, però a un cost d’administració alt, ja que hem de disposar de coneixements amplis dels dispositius i de les característiques que estem escollint (què signifiquen i quines implicacions poden tenir), i també de les conseqüències que puguin tenir les decisions que prenguem.

3. Procés de configuració i compilació

La personalització del nucli [Vasb] és un procés costós, necessita amplis coneixements del procés que s'ha de dur a terme i, a més, és una de les tasques crítiques, de la qual depèn l'estabilitat del sistema, per la mateixa naturalesa del nucli, ja que n'és, per al sistema operatiu, l'element central.

Qualsevol error de procediment pot comportar la inestabilitat o la pèrdua del sistema. Per tant, no és sobrer fer tasques de còpia de seguretat de les dades d'usuaris, dades de configuracions que hàgim personalitzat o, si disposem de dispositius adequats, fer una còpia de seguretat completa del sistema. També és recomanable disposar d'algun disquet d'arrencada (o distribució *live CD* amb eines de rescat) que ens serveixi d'ajuda si sorgeixen problemes, o bé un disquet/CD/arxiu de rescat (*rescue disk*) que la majoria de distribucions permeten crear des dels CD de la distribució (o en proporcionen directament algun com a CD de rescat per a la distribució). Actualment molts dels CD autònoms (*live CD*) de les distribucions ja proporcionen prou eines de rescat per a aquestes tasques, tot i que també hi ha algunes distribucions que hi estan especialitzades.

No obstant això, davant sistemes en producció, sempre és important prendre les mesures de precaució i fer les còpies de seguretat necessàries. O bé treballar amb un sistema equivalent, en estat de test o preproducció, en el qual puguem prèviament provar els efectes d'un nou kernel.

Examinem el procés necessari per a instal·lar i configurar un nucli Linux. En els subapartats següents, veurem:

- 1) El procés de compilació per a les branques 2.6.x i 3.x
- 2) Detalls específics de les versions 3.x.
- 3) Un cas particular per a la distribució Debian, que disposa d'un sistema propi (*Debian Way*) de compilació més flexible.

Respecte a les versions 2.6.x, en aquest mòdul mantenim l'explicació per raons històriques, ja que les distribucions actuals pràcticament ja no les ofereixen, però hem de considerar que en més d'una ocasió ens veurem obligats a migrar un determinat sistema a noves versions, o bé a mantenir-lo en les antigues per incompatibilitats o per l'existència de maquinari antic no suportat per les distribucions actuals.

Obtenció d'un nucli personalitzat

El procés d'obtenció d'un nou nucli personalitzat passa per obtenir els paquets font, adaptar la configuració, compilar i instal·lar el nucli obtingut en el sistema.

Enllaç d'interès

Per a Fedora, recomanem consultar l'enllaç següent: http://fedoraproject.org/wiki/Building_a_custom_kernel.

Els conceptes generals del procés de compilació i configuració s'explicaran en el primer subapartat (2.6.x), ja que la majoria són genèrics, i observarem posteriorment les diferències respecte a les noves versions.

També cal afegir que, amb les últimes distribucions, cada vegada és més casual la necessitat de reconstruir o recompilar el nucli mateix, a causa, entre altres consideracions, dels següents fets:

- Antigament la majoria de controladors estaven integrats en el nucli i calia compilar-lo per complet si volíem incloure o excloure un controlador determinat. Avui dia, com veurem en l'apartat 5, poden compilar-se els controladors o mòduls concrets, no el nucli en si mateix.
- Per a sintonitzar el nucli, antigament calia compilar-lo. En molts casos (no tots) es poden sintonitzar alguns elements del nucli mitjançant l'accés als sistemes de fitxers `/proc` o `/sys`.
- En alguns casos de distribucions comercials (versions empresarials per a les quals es paga suport), els nuclis i el sistema complet estan suportats per l'equip de la distribució, i de vegades poden perdre's les llicències de suport o les garanties per a fer canvis d'aquest estil.
- D'altra banda, les distribucions tenen una velocitat bastant ràpida quant a integrar pedaços i nous nuclis a mesura que es generen.

En canvi, una personalització del nucli, mitjançant compilació, ens pot permetre:

- Escollir quines parts incloure o excloure del nucli, donant un suport concret a una plataforma o a un maquinari molt concret. Això és imprescindible si ens trobem, per exemple, en situacions de maquinari encastat (*embedded*).
- Sintonitzar, d'aquesta última manera, el consum de memòria o altres recursos per a adaptar-se millor a recursos limitats (CPU, memòria, disc, etc.).
- Versions concretes de les distribucions impliquen usar certes versions específiques del nucli. En molts casos no podem obtenir noves actualitzacions del nucli si no actualitzem la distribució concreta completament a una nova *release* d'aquesta.
- Provar versions de nucli encara no disponibles en les distribucions. Malgrat que hi ha certa rapidesa d'integració dels nous nuclis, es pot trigar setmanes o mesos a disposar dels nous nuclis via distribució. D'altra banda, algunes distribucions són molt conservadores quant a la versió de nucli utilitzada, i cal esperar diverses versions completes de la distribució (un període llarg de temps, per exemple sis mesos) per a arribar a una versió concreta de nucli.

- Provar versions beta, o pedaços de nucli, amb l'objectiu d'integrar-lo ràpidament en algun sistema amb problemes concrets, o bé senzillament per qüestions d'avaluació de les noves possibilitats o d'un millor rendiment.
- Participar directament en el desenvolupament del nucli, estudiant possibles millores del codi actual, proposant i provant propostes concretes. Això és típic d'alguns components, i també estudiar diferents estratègies de planificació de CPU i de gestió de memòria, millorar paràmetres del nucli o col·locar alguna prestació nova en algun sistema de fitxers.

En els subapartats següents veurem les diferents possibilitats quant a la configuració i la compilació de les diferents branques de desenvolupament del nucli Linux.

3.1. Compilació de les branques 2.6.x i 3.x del kernel Linux

Les instruccions són específiques per a l'arquitectura x86/x86_64 (o amd64) d'Intel, mitjançant usuari *no-root* en la major part del procés (es pot fer com a usuari normal i, de fet, és altament aconsellable per seguretat), i només al procés final d'instal·lació és necessari *root* per a integrar el kernel final en el sistema. La compilació del kernel és un procés costós en temps i disc per als kernels actuals, i es recomana disposar d'espai de disc abundant (100 GB, o més, per a kernels actuals), i portar a terme el procés amb el màxim de CPU i memòria disponible per a accelerar-lo. Les diferents fases que hi estan involucrades són:

1) Obtenir el nucli. Podem acudir a <http://www.kernel.org> (o al seu servidor ftp) i descarregar la versió *stable* o *longterm* que volem provar. D'altra banda, en la majoria de distribucions de GNU/Linux, com ara Fedora / Red Hat o Debian, també s'ofereix com a paquet el codi font del nucli (normalment amb algunes modificacions incloses); si es tracta de la versió del nucli que necessitem, potser és preferible fer servir aquestes (mitjançant els paquets `kernel-source`, `kernel-version`, `linux-source`, o similars). Si volem els últims nuclis, potser no estaran disponibles en la distribució i haurem d'acudir a *kernel.org*.

2) Desempaquetar el nucli. Els paquets font del nucli acostumaven a col·locar-se i desempaquetar-se sobre el directori `/usr/src`, tot i que es recomana fer servir algun directori a part per a no barrejar-los amb fitxers font que pugui portar la distribució. Per exemple, si els paquets font venien en un fitxer comprimit de tipus bzip2:

```
bzip2 -dc linux-2.6.32.63.tar.bz2 | tar xvf -
```

Si els paquets font venien en un fitxer gz (o `tar.gz`), reemplacem `bzip2` per `gzip`, o en els recents (fets servir per kernel.org) `.tar.xz` o `.xz` reemplacem per `unxz` (paquet `xz-utils`). En descomprimir els paquets font, s'haurà generat un directori denominat `linux-version_kernel`, on entrarem per establir la configuració del nucli.

En aquest punt, podria ser interessant apedaçar el nucli, la qual cosa podríem fer ja que tenim un codi font addicional (en forma de pedaç) que millora algun problema conegut de la versió o bé perquè volem proposar o provar un canvi de codi en el nucli. També es podria donar el cas que un fabricant de maquinari oferís algun suport o correcció de fallades per a un controlador de dispositiu, com un pedaç per a una versió de nucli concreta.

Una vegada disposem dels paquets necessaris, procedim al procés de compilació en el directori fet servir pels paquets font. Cal recordar que tot el procés es pot portar a terme com a usuari normal; només parts molt concretes, com la instal·lació final del nucli o de mòduls dinàmics, és necessari fer-les fent servir l'usuari *root*.

També poden sorgir problemes de seguretat per a fer la compilació en mode *root* de fonts de nucli desconegudes o no fiables. Respecte a això, tant els paquets font de *kernel.org* com els proporcionats per les distribucions acostumen a contenir firmes (o repositoris firmats) que es poden fer servir per a verificar la integritat dels fitxers de fonts. S'ha d'evitar, costí el que costí, fer servir fonts de nucli o de mòduls proporcionats per emissors no fiables.

Vegeu també

El procés d'apedaçar el nucli es veu en l'apartat 4.

Eines de configuració i compilació

Abans de començar els passos previs a la compilació, hem d'assegurar-nos de disposar de les eines correctes, especialment del compilador gcc, make i altres utilitats GNU complementàries en el procés. Un exemple són les *module-init-tools*, que ofereixen les diferents utilitats per a l'ús i gestió dels mòduls de nucli dinàmics. Així mateix, per a les diferents opcions de configuració cal tenir en compte una sèrie de prerequisits en forma de biblioteques associades a la interfície de configuració utilitzada (per exemple, les ncurses per a la interfície menuconfig).

Per exemple, en una distribució Debian estable (en altres distribucions o versions, haureu de consultar els paquets similars), són necessaris, entre d'altres, els paquets *build-essential libncurses5-dev libqt4-dev libqt4-qmake libgtk2.0-dev libglib2.0-dev libglade2-dev modules-init-tools gcc g++*. En el cas de Debian, necessitem una instal·lació prèvia d'aquests paquets, per exemple, feta amb la llista anterior, i un *apt-get install* dels paquets esmentats.

Es recomana, en general, consultar la documentació del nucli (per mitjà del paquet o en el directori arrel de les fonts del nucli) per a saber quins prerequisits, i versions d'aquests, són necessaris per al procés. Es recomana examinar els fitxers *README* en el directori "arrel", i el *Documentation/Changes*, que esmenta els requisits mínims de versions dels programes, prerequisits de la compilació, o l'índex de documentació del nucli en *Documentation/00-INDEX*.

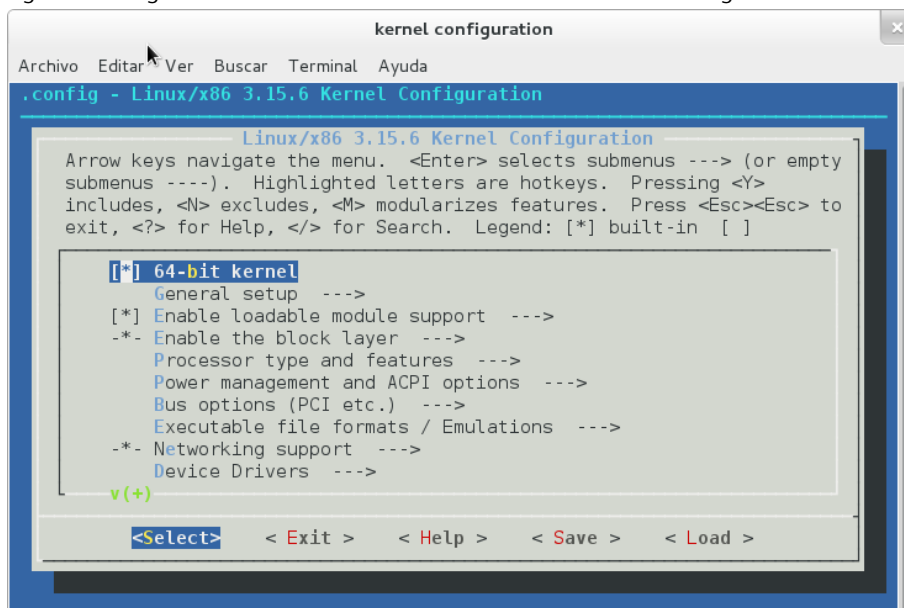
Si hem fet anteriors compilacions en el mateix directori, hem de garantir que el directori utilitzat estigui net de compilacions anteriors; podem netejar-lo amb *make mrproper* (fet des del directori arrel de les fonts).

3) Configuració del nucli. Per al procés de configuració del nucli [Vasb], tenim diversos mètodes alternatius, que ens presenten interfícies diferents per a ajustar els múltiples paràmetres del nucli (que s'acostumen a emmagatzemar en un fitxer de configuració, normalment *.config* en el directori "arrel" de les fonts). Les diferents alternatives són:

- *make config*: des de la línia d'ordres, se'ns pregunta per cada opció i se'ns demana confirmació (*y/n*), si desitgem o no l'opció, o se'ns demanen els valors necessaris, o seleccionar una de les possibilitats de l'opció. És la configuració llarga, en la qual se'ns demanen moltes respostes; podem haver de respondre uns quants centenars de preguntes (o més, depenent de la versió).

- `make oldconfig`: serveix per si volem reutilitzar una configuració ja emprada (normalment emmagatzemada en un fitxer `.config`, en el directori “arrel” de les fonts prèvies que hàgim completat amb anterioritat). Cal tenir en compte que només és vàlida si estem compilant la mateixa versió del nucli, ja que diferents versions del nucli poden variar en les seves opcions. No obstant això, en les últimes versions del nucli, aquesta opció detecta la configuració anterior i només ens pregunta per les opcions noves disponibles en la nova versió, estalviant així un temps considerable en les eleccions. D'altra banda, en general la configuració d'un kernel disponible en la distribució està també guardada en el fitxer `/boot/config-version` corresponent (recordeu que amb `uname -r` tenim la versió del kernel actual), el qual podem copiar directament com a fitxer `.config` en l'arrel de les fonts del kernel, i amb `make oldconfig` configurar només les noves opcions.
- `make menuconfig`: configuració basada en menús textuais, molt còmoda (figura 2); podem habilitar o inhabilitar el que vulguem, i és més ràpida i versàtil per a la navegació entre opcions que el `make config`.
- `make xconfig`: la més còmoda, basada en diàlegs gràfics en X Window. És necessari tenir instal·lades les biblioteques de desenvolupament de qt3 o qt4, ja que aquesta configuració té aquests requeriments en estar basada en la biblioteca Qt3/Qt4 (KDE). La configuració es basa en quadres de diàleg, botons i caselles d'activació. És prou ràpida i disposa d'ajuda amb comentaris de moltes de les opcions.
- `make gconfig`: similar al cas de `xconfig`, però amb la interfície basada en gtk (Gnome). Com a prerequisit, necessita els paquets de desenvolupament de les biblioteques Gnome (entre d'altres, *libgtk2.0-dev*, *libglib2.0-dev* i *libglade2-dev*; els noms són dependents de la distribució i versió).

Figura 2. Configuració del nucli 3.x des de la interfície textual de menuconfig



Un cop s'hagi fet el procés de configuració, cal guardar el fitxer `.config`, ja que la configuració consumeix un temps important. A més, pot ser d'utilitat disposar de la configuració efectuada (`.config`) si s'està planejant fer-la posteriorment en diferents màquines similars o idèntiques.

Un altre tema important en les opcions de configuració és que en molts casos se'ns preguntarà si una determinada característica la volem integrada en el nucli o com a mòdul. Aquesta és una decisió més o menys important, ja que la mida i el rendiment del nucli (i, per tant, del sistema sencer) en alguns casos pot dependre de la nostra elecció.

Vegeu també

La gestió de mòduls es veu en l'apartat 5.

El nucli de Linux, com a imatge binària un cop compilat, ha començat a ser molt gran, tant per complexitat com pels controladors de dispositiu [Cor05] que inclou. Si hi integréssim totes les seves possibilitats, es podria crear un fitxer del nucli prou gran i ocupar molta memòria, fet que alentiria alguns aspectes de funcionament. Els mòduls del nucli [Hen] són un mètode que permet separar part del nucli en petites porcions, que s'encarregaran de manera dinàmica a demanda quan, per càrrega explícita o per l'ús de les seves característiques, siguin necessàries.

L'elecció més normal és integrar dins del nucli allò que es consideri bàsic per al funcionament, o crític en rendiment, i deixar com a mòduls aquelles parts o controladors dels quals es faci un ús esporàdic o que es necessitin conservar per si es produeixen futures ampliacions de l'equip.

- Un cas clar són els controladors de dispositiu: si estem actualitzant la màquina, pot ser que a l'hora de crear el nucli no sapiguem amb seguretat quin maquinari tindrà (per exemple, quina targeta de xarxa), però sí sabem que estarà connectada a xarxa; en aquest cas, el suport de xarxa estarà integrat en el nucli, però pel que fa als controladors de les targetes, podem seleccionar-ne alguns (o tots) i posar-los com a mòduls. D'aquesta manera, quan tinguem la targeta podrem carregar el mòdul necessari, o si després hem de canviar una targeta per una altra, només haurem de canviar el mòdul que es carregarà. Si només hi hagués un controlador integrat en el nucli i canviem la targeta, això obligaria a reconfigurar i recompilar el nucli amb el controlador de la nova targeta.
- Un altre cas que acostuma a aparèixer (tot i que no és gaire comú) té lloc quan necessitem dos dispositius que són incompatibles entre ells, o està funcionant l'un o l'altre (això passava antigament, per exemple, amb impressores amb cable paral·lel i alguns dispositius de maquinari que es connectaven al port paral·lel). Per tant, en aquest cas hem de col·locar com a mòduls els controladors i carregar o descarregar allò que sigui necessari en cada moment. Tot i que actualment també s'han solucionat aquests problemes, amb suports de *daemons* que controlen casos de *hotplug* (connexió en calent) de dispositius, com per exemple *udev*, que aporta solucions en aquest sentit ja que gestiona dinàmicament les entrades del directori `/dev` a mesura que els dispositius es connecten o desconnecten en el sistema.

- Un altre exemple el podrien formar els sistemes de fitxers (*filesystems*). Normalment, esperarem que el nostre sistema tingui accés a alguns d'aquests, per exemple ext2, ext3 o ext4 (propis de Linux), vfat (dels Windows 95/98/ME), i els donarem d'alta en la configuració del nucli. Si en un altre moment haguéssim de llegir un altre tipus no esperat, per exemple dades guardades en un disc o partició de sistema NTFS de Windows NT/XP, no podríem: el nucli no sabria o no tindria suport per a fer-ho. Si tenim previst que en algun moment (però no habitualment) s'hagi d'accedir a aquests sistemes, podem deixar els altres sistemes de fitxers com a mòduls.

4) Compilació del nucli. Un cop disponible una configuració de la versió del nucli, mitjançant `make` començarem el procés de compilació:

```
$ make
```

Aquest procés es pot accelerar si disposem d'una màquina multiprocessador o *multicore*; `make` disposa de l'opció `-jn`, amb la qual podem especificar, amb `n`, el nombre de processadors o *cores* que farem servir per a la compilació.

Quan aquest procés finalitzi, tindrem la part integral del nucli i ens faltaran les parts que es va demanar col·locar com a mòduls:

```
$ make modules
```

Fins a aquest moment, hem fet la configuració i compilació del nucli. Aquesta part es podia fer des d'un usuari normal (altament recomanable) o bé el *root*, però ara necessitarem forçosament usuari *root*, perquè passarem a la part de la instal·lació del nou kernel compilat en el sistema.

5) Instal·lació. Comencem instal·lant els mòduls:

```
# make modules_install
```

Això ens instal·larà els mòduls construïts en el sistema de fitxer perquè el nou kernel els pugui trobar al lloc adequat. Normalment, els mòduls acostumen a estar disponibles en el directori `/lib/modules/version_kernel`, on `version_kernel` serà la numeració del kernel que acabem de construir.

I per a realitzar la instal·lació del nou nucli (des del directori de compilació `dir/linux-version`, essent `version` la versió emprada), simplement amb:

```
# make install
```

Per a aquest últim pas, la majoria de distribucions inclouen un suport extern amb un *script* denominat *installkernel* (normalment proporcionat pel paquet *mkinitrd*), que el fa servir el sistema de compilació del kernel per a col·locarlo al lloc adequat, i modificar el *bootloader* (LILO o Grub), de manera que no s'hagin de fer passos extra. Cal indicar que algunes distribucions de tipus *from*

Cas Debian

Algunes distribucions permeten modes més simplificats; per exemple, Debian permet mitjançant *make deb-pkg* portar a terme tot el procés de compilació i preparar la instal·lació mitjançant la creació dels paquets `.deb` per a la instal·lació del kernel. Finalment, només quedarà instal·lar amb *dpkg -i* els paquets resultants.

scratch, com *Gentoo*, no inclouen l'script, i per tant cal consultar la documentació específica de kernel per a aquestes distribucions.

En el procés d'aquesta última fase *install*, es desenvolupen les fases següents:

- Es verifica que el kernel hagi estat ben construït.
- S'instal·la la porció estàtica del kernel en el directori `/boot`, i posa nom al fitxer amb la versió del kernel que s'ha construït (típicament haurà de ser `/boot/vmlinuz-version`).
- Les imatges de ramdisk que puguin ser necessàries són creades fent servir aquells mòduls que s'hagin creat de manera prèvia i siguin necessaris per a aquestes imatges quan arrenqui la màquina. Típicament, aquesta imatge apareixerà com a `/boot/initrd.img-version`. Respecte a la creació de *initrd*, cal disposar de les eines adequades; en Debian, per exemple, en el procés de compilació és necessari com a prerequisit el paquet *initramfs-tools* (per a kernels superiors a 2.6.12, antigament complien aquesta funció les *mkinitrd-tools*, ja obsoletes). També durant aquestes primeres fases s'acostumen a generar el fitxer `/boot/System.map-version` i el fitxer `/boot/config-version`, que contenen, de manera respectiva, informació sobre els símbols disponibles en el kernel i la configuració feta servir en el kernel en la seva compilació.
- El *bootloader* (LILO/Grub) corresponent del sistema rep la notificació de l'existència del nou kernel, i es crea l'entrada necessària del menú, de manera que l'usuari pugui seleccionar-lo en la següent arrencada de la màquina.

Finalment, després d'aquestes fases, el kernel està instal·lat correctament i podem procedir al reinici, i provar la nostra imatge nova del kernel. Si el procés s'ha fet correctament, disposarem d'una entrada de *bootloader* corresponent i, en cas que sorgissin problemes amb la nova imatge, disposaríem de les entrades dels kernels antics per a tornar a la situació estable anterior.

En la instal·lació, depenent del kernel i del seu ús posterior, també pot ser necessària la instal·lació dels *headers* de desenvolupament del kernel, i dels *firmwares* associats (els últims kernels han inclòs alguns *firmwares* de dispositius directament en el kernel). Aquests últims processos els desenvoluparem amb:

```
# make headers_install
# make firmware_install
```

I, d'aquesta manera, finalitzarem el procés d'instal·lació del kernel complet.

Aquest últim procés, d'instal·lació automàtica mitjançant *make install* (i afegits), també es pot fer de manera manual, si no es disposa de l'*script* abans comentat (installkernel) amb el següent procés (alguns detalls depenen de l'arquitectura):

```
# make modules install
# make kernelversion
```

(l'última ordre obté un número `KERNEL_VERSION` de la versió construïda, en el que segueix s'ha de col·locar el número obtingut on aparegui `KERNEL_VERSION`)

```
# cp arch/i386/boot/bzImage /boot/bzImage-KERNEL_VERSION
# cp System.map /boot/System.map-KERNEL_VERSION
```

L'arxiu `bzImage` és el nucli acabat de compilar, que es col·loca en el directori `/boot`. Normalment, el nucli antic es trobarà en el mateix directori `boot` amb el nom `vmlinuz` o bé `vmlinuz-versión-anterior` i `vmlinuz` com un enllaç simbòlic al nucli antic. Una vegada tinguem el nostre nucli, és millor conservar l'antic, per si es produeixen fallades o un mal funcionament del nou i així poder recuperar el vell. El fitxer `System.map`, fitxer que conté els símbols disponibles en el nucli, necessari per al procés d'arrencada del nucli, també es col·loca en el mateix directori `/boot`.

També pot ser necessària la creació de la imatge de disc ramdisk `initrd` amb les utilitats necessàries segons la distribució (o la seva versió).

Respecte a la creació del `initrd`, en Fedora / Red Hat aquest es crearà de manera automàtica amb l'opció `make install`. En Debian haurem de fer servir les tècniques conegudes com a *Debian Way* o bé crear-lo de manera explícita amb `mkinitrd` (versions de nucli $\leq 2.6.12$) o, posteriorment, amb kernels actuals utilitzar, bé `mkinitramfs`, o una eina denominada `update-initramfs`, indicant la versió del nucli (s'assumeix que aquest es diu `vmlinuz-version` en el directori `/boot`):

```
# update-initramfs -c -k 'version'
```

Tot i que cal indicar que en les distribucions actuals, i versions del kernel, és habitual que el `initramfs` es creï sol en el procés de compilació i posterior instal·lació (en el *make install* igualment). Tot i així, es recomana la remodelació d'aquest (via prèvia configuració en `/etc/initramfs-tools/initramfs.conf`), ja que acostuma a presentar una mida respectable, que pot ser ineficient en algunes màquines perquè inclou més mòduls dels imprescindibles. Per a això, es pot jugar (en el mencionat `initramfs.conf`) amb la configuració `MODULES=most` o `dep` (que segurament minimitzarà la mida de manera significativa). Després, podem recrear el `initrd` mitjançant l'ordre anterior o, per exemple, substituint els paràmetres per:

```
#update-initramfs -t -u -k version
```

que en aquest cas ens actualitzarà el initrd si el procés d'instal·lació ja en va crear un de previ.

6) Configuració de l'arrencada. El següent pas és dir al sistema amb quin nucli ha d'arrencar. Aquest pas depèn del sistema d'arrencada de Linux, i del *bootloader* emprat:

- Si s'arrenca amb LILO [Skoa], ja sigui en el Master Boot Record (MBR) o des de partició pròpia, cal afegir-hi el fitxer de configuració (localitzat en `/etc/lilo.conf`), per exemple, les línies:

```
image = /boot/bzImage-KERNEL_VERSION
label = KERNEL_VERSION
```

on `image` és el nucli que s'arrencarà i `label` serà el nom amb què apareixerà l'opció en l'arrencada. Podem afegir-hi aquestes línies o modificar les que hi hagués del nucli antic. Es recomana afegir-les i deixar el nucli antic per a recuperar-lo si hi ha problemes. En el fitxer `/etc/lilo.conf` hi pot haver una o més configuracions d'arrencada, tant de Linux com d'altres sistemes (com Windows); cada arrencada s'identifica per la seva línia `image` i el `label` que apareix en el menú d'arrencada. Hi ha una línia `"default=label"` on s'indica el `label` per defecte que s'arrencarà. També podem afegir a les línies anteriors un `"root=/dev/..."` per a indicar la partició de disc on estarà el sistema d'arxius principal (el `/`). Recordeu que els discos tenen dispositius com ara `/dev/sda` (primer disc) `/dev/sdb` (segon), i la participació s'indicaria com `"root=/dev/sda2"` si el `/` del nostre Linux estigués en la segona partició del primer disc. A més a més, amb `append=` podem afegir paràmetres a l'arrencada del nucli. Després de canviar la configuració del LILO, cal escriure-la físicament en el disc (es modifica el sector d'arrencada) perquè estigui disponible en la següent arrencada:

```
/sbin/lilo -v
```

- Arrencada amb Grub: en aquest cas, la gestió és molt simple; cal introduir una nova configuració formada pel nucli nou i afegir-la com una opció més en el fitxer de configuració del Grub, i tornar a arrencar procedint de manera semblant a la del LILO, però recordant que en Grub n'hi ha prou d'editar el fitxer i tornar a arrencar. També és millor deixar l'antiga configuració per a recuperar-se de possibles errors o problemes amb el nucli acabat de compilar. Cal assenyalar que la configuració de Grub (normalment disponible en `/boot/grub/menu.lst` o `grub.cfg`) depèn molt de la versió, tant si és Grub-Legacy (Grub 1.x) com Grub 2. Es recomana examinar la configuració de kernels ja presents, per a adaptar l'acabat de crear, i consultar la documentació GNU i el manual disponible via *info grub*.

Lectura recomanada

Sobre Grub, podeu consultar *Grub bootloader* i *Grub Manual* accessibles des del web del projecte GNU: <http://www.gnu.org>

Un cop disposem dels fitxers correctes en `/boot` i del *bootloader* actualitzat, ja podem procedir a tornar a arrencar amb `shutdown -r now`, escollir el nostre nucli i, si tot ha anat bé, podem comprovar amb `uname -r` que disposem de la nova versió del nucli. També és particularment interessant examinar alguns registres, com ara `/var/log/messages` (o *logs* equivalents, depenent de la distribució) i l'ordre `dmesg`, per a examinar el registre de sortida de missatges produïts pel nou nucli en l'arrencada i detectar si ha sorgit algun problema de funcionalitat o amb algun dispositiu concret.

Si tinguéssim problemes, podem recuperar l'antic nucli, escollir l'opció del vell nucli i després retocar el *bootloader* per a tornar a l'antiga configuració o estudiar el problema i reconfigurar i compilar el nucli una altra vegada.

3.2. Compilació del nucli en Debian (*Debian Way*)

En Debian, a més dels mètodes comentats en els subapartats previs, cal afegir la configuració pel mètode denominat *Debian Way*. Es tracta d'un mètode que ens permet construir el nucli d'una manera flexible i ràpida, adaptada a la distribució.

Per al procés, necessitarem una sèrie d'utilitats (cal instal·lar els paquets o similars): `kernel-package`, `ncurses-dev`, `fakeroot`, `wget` i `bzip2`.

Podem observar el mètode [Debk] des de dues perspectives: reconstruir un nucli equivalent al proporcionat per la distribució com a nucli base (canviant opcions) o bé crear un nucli amb una numeració de versió-revisió personalitzada.

Pel que fa als paquets de fonts del nucli, Debian proporciona els paquets font utilitzats en la seva distribució, que poden arribar a ser molt diferents dels de la versió *vanilla* o *pristine* obtinguda com a estable de *kernel.org*. Això es deu al fet que en Debian es produeixen múltiples revisions amb diversos pedaços que s'hi van afegint, molts d'aquests a partir de fallades que es detecten *a posteriori* en les següents versions *vanilla* del nucli.

En les versions estables de Debian, la distribució sol escollir una revisió `xx` de la branca `2.6.xx` o `3.xx`, de manera que el nucli acostuma a quedar-se (generalment) en aquesta numeració per a tota la vida de la versió concreta de Debian estable i, així, quan s'actualitza amb revisions menors la distribució, només s'actualitzen pedaços del nucli (sense canviar el número principal). Quan Debian produeix la següent versió estable, se salta a una nova versió del nucli. Mentre dura una versió estable de la distribució, Debian sol produir diferents modificacions (*patchlevels*) o revisions del nucli que es va escollir.

Debian ha canviat diverses vegades la gestió dels seus paquets associats als paquets font del nucli. A partir de la versió 2.6.12, és habitual trobar en el

Enllaç d'interès

Per a Fedora, recomanem consultar el següent enllaç:
http://fedoraproject.org/wiki/Building_a_custom_kernel.

Enllaç d'interès

Es pot veure el procés *Debian Way* de manera detallada en:
<http://kernel-handbook.alioth.debian.org/>

repositori Debian una versió `linux-source-version` que conté la versió de les fonts del nucli amb els últims pedaços aplicats (vegeu l'apartat 4). Aquesta versió del paquet de les fonts del nucli és la que farem servir per a crear un nucli personalitzat, en l'esmentada *Debian Way*. Aquest paquet font és usat per a crear els paquets binaris de la distribució associats al nucli i també és l'indicat per a fer servir en cas de voler aplicar pedaços al nucli actual de la distribució, o per si volem provar modificacions del nucli en un nivell de codi.

Examinem primer aquesta opció i després, al final del subapartat, comentarem la personalització.

Per a la reconstrucció del kernel actual Debian, podem procedir de dues maneres: o bé obtenim el kernel directament de les fonts (amb tots els pedaços inclosos), o bé reconstruïm els paquets Debian oficials del kernel.

En el primer cas, obtenim les fonts directes i per a la compilació del nucli actual procedim usant el paquet disponible com `linux-source-version`; en aquest cas d'exemple `version=3.2` (versió principal del kernel per a una Debian estable concreta) com a versió del kernel Debian de la distribució, però depenent en cada moment del kernel actual, haurem d'escollir les fonts que es corresponguin amb la versió principal del nostre kernel actual (vegeu `uname -r`):

```
# apt-get install linux-source-3.2
$ tar jxf /usr/src/linux-source-3.2.tar.bz2
```

En aquest cas les descarregarà normalment en `/usr/src`, encara que podem copiar-les a un altre espai o disc de treball, ja que el procés necessita un espai important (prop de 10 GB o més), si no disposem d'aquest a `/usr/src`. Una vegada tinguem la font, amb `tar`, descomprimirà els paquets font i els deixarà en un arbre de directoris a partir del directori `linux-version`, on la versió ara serà la revisió del kernel disponible (3.2.xx). Per a la configuració i compilació posterior a partir d'aquest directori, cal fer servir el mètode clàssic o la *Debian Way* que examinem en aquest apartat.

Per al segon cas (la reconstrucció dels paquets Debian oficials), l'objectiu és obtenir uns paquets *deb* finals equivalents als que ofereix la distribució, però amb la personalització que vulguem.

Per a aquest procés, començarem per obtenir algunes eines que necessitem:

```
# apt-get install build-essential fakeroot
# apt-get build-dep linux
```

Aquestes línies bàsicament ens instal·len l'entorn de compilació per al nucli (de fet, els paquets del compilador `gcc` necessaris i eines pròpies de la *Debian*

Way) i finalment es comproven les dependències de les fonts per si són necessaris paquets font extra de desenvolupament. Després d'aquesta configuració d'eines inicials, podem passar a la descàrrega de les fonts (es pot fer des de qualsevol usuari, no és necessari *root* per al procés):

```
$ apt-get source linux
```

que ens descarregarà les fonts i les descomprimirà en el directori actual, a més d'altres paquets que inclouen pedaços respecte a la versió original del kernel. El directori necessari on començar el procés el trobarem com `linux-version`. En aquest moment, podrem personalitzar la configuració del kernel pels mètodes examinats en la secció prèvia, ja que la configuració és la que porta el kernel Debian per defecte, de manera que podrem canviar aquells components o opcions del kernel que ens interessin abans de tornar a reconstruir els paquets.

Per a portar a terme la construcció dels paquets kernel per a l'arquitectura (es recomanen com a mínim 20 GB disponibles de disc), segons la configuració preestablerta del paquet (semblant a la inclosa en els paquets oficials del nucli `linux-image` en Debian):

```
$ cd linux-version
$ fakeroot debian/rules binary
```

Amb això disposarem dels paquets binaris (i uns altres de suport independents de l'arquitectura) del nucli (fitxers `*.deb`) disponibles per a la instal·lació (via `dpkg -i`).

Hi ha alguns procediments extra per a la creació del nucli sobre la base de diferents nivells de pedaça (*patch*) proporcionats per la distribució, i possibilitats de generar diferents configuracions finals (es pot veure la referència de la nota per a complementar aquests aspectes).

Passem ara a l'altra opció que havíem comentat al principi, als aspectes de personalització, quan volem canviar certes opcions del nucli i crear-ne una versió personal (a la qual donarem una numeració de revisió pròpia).

En aquest cas, més habitual, quan desitgem un nucli personalitzat, haurem de seguir un procés semblant mitjançant un pas de personalització típic (per exemple, mitjançant `make menuconfig` o `oldconfig`). Els passos són, en primer lloc, l'obtenció i preparació del directori (aquí obtenim els paquets de la distribució, però és equivalent obtenint els paquets font des de *kernel.org*). En aquest cas sigui 3.x la versió de kernel disponible de fonts en el repositori, o també començant en el segon pas, el kernel estable obtingut de *kernel.org* (aquest últim potser estigui en format *xz* en lloc de *bzip2*):


```
# apt-get install linux-source-3.2
$ tar -xvjf /usr/src/linux-source-3.2.tar.bz2
$ cd linux-source-3.2
```

d'on obtenim els paquets font i els descomprimim (la instal·lació del paquet deixa l'arxiu de fonts en `/usr/src`; en cas que procedeixi de `kernel.org`, ja depèn d'on haguem fet la descàrrega inicial). És recomanable que abans de procedir al pas de descompressió amb `tar`, el portem a terme en un espai de disc amb capacitat suficient (recomanats 10-20 GB).

A continuació, fem la configuració de paràmetres. Com sempre, podem basarnos en fitxers `.config` que hem utilitzat anteriorment per a partir d'una configuració coneguda (per a la personalització, amb *make oldconfig*, també es pot fer servir qualsevol dels altres mètodes, *xconfig*, *gconfig*, etc.):

```
$ make menuconfig
```

El kernel preparat d'aquesta manera, tret que indiquem el contrari, té la depuració activada, la qual cosa resulta molt útil per a la depuració en cas d'errors (utilitzant eines com *kdump*, *crash* o *SystemTap*), encara que, d'altra banda, ens augmenta l'espai necessari del kernel per a la seva construcció; quan això no sigui necessari, abans de passar a la construcció podem deshabilitar l'opció d'incloure informació de depuració:

```
$ scripts/config --disable DEBUG_INFO
```

A continuació, la construcció final del nucli:

```
$ make clean
$ make KDEB_PKGVERSION=custom.1.0 deb-pkg
```

on vam crear un identificador per al nucli construït (`custom.1.0`) que s'afegirà al nom del paquet binari del nucli, posteriorment visible en l'arrencada amb l'ordre `uname`.

Així, el procés finalitzarà amb l'obtenció dels paquets associats a la imatge del nucli, que podrem finalment instal·lar (examinar dels passos anteriors el número del paquet obtingut, que serà el que inclourem aquí):

```
# dpkg -i ../linux-image-3.xx.yy_custom.1.0_i386.deb
```

Això ens descomprimirà i instal·larà el nucli i generarà una imatge `initrd` addicional si és necessari. A més, ens configura el *bootloader* amb el nou nucli

per defecte (cal vigilar amb aquest pas: val la pena haver obtingut abans una còpia de seguretat del *bootloader* per a no perdre cap configuració estable).

Ara, directament amb un `shutdown -r now` podem provar l'arrencada amb el nou nucli.

Afegim també una altra peculiaritat que cal tenir en compte en Debian, que pot ser d'utilitat amb algun maquinari: l'existència d'utilitats per a afegir mòduls dinàmics de nucli proporcionats per tercers; en particular, la utilitat `module-assistant`, que permet automatitzar tot aquest procés a partir dels paquets font del mòdul.

Necessitem disposar dels *headers* del nucli instal·lat (disponible en el paquet `linux-headers-version`) o bé dels paquets font que utilitzem en la seva compilació (també es poden obtenir durant la instal·lació del nucli mitjançant un `make headers_install`, o mitjançant el paquet `deb` obtingut amb *Debian Way*). A partir d'aquí, `module-assistant` es pot utilitzar interactivament i seleccionar entre una àmplia llista de mòduls registrats prèviament en l'aplicació, i pot encarregar-se de descarregar el mòdul, compilar-lo i instal·lar-lo en el nucli existent.

En la utilització des de la línia d'ordres, també podem simplement especificar (`m-a`, equivalent a `module-assistant`):

```
# m-a prepare
# m-a auto-install nom_modul
```

Així es prepara el sistema per a possibles dependències, descarrega fonts del mòdul, compila i, si no hi ha problemes, instal·la per al present nucli. En la llista interactiva de `module-assistant`, podem observar el nom dels mòduls disponibles.

Enllaç d'interès

Habitualment no serà necessari, però si es necessita alguna reconfiguració del `initrd` generat, es recomana llegir el següent enllaç, on es comenten algunes eines Debian disponibles:
<http://kernel-handbook.alioth.debian.org/ch-initramfs.html>

4. Aplicació de pedaços (*patch*) al nucli

En alguns casos, també pot ser útil l'aplicació de pedaços (*patches*) al nucli [Lkm].

Un fitxer de pedaç (*patch file*) respecte al nucli de Linux és un fitxer de text ASCII que conté les diferències entre el codi font original i el nou codi, amb informació addicional de noms de fitxer i línies de codi. El programa *patch* (vegeu `man patch`) serveix per a aplicar-lo a l'arbre del codi font del nucli (normalment, depenent de la distribució, en `/usr/src/linux`).

Els pedaços s'acostumen a necessitar quan un maquinari especial necessita alguna modificació en el nucli, o s'han detectat alguns errors (*bugs*) de funcionalitat posteriors a alguna distribució concreta d'una versió del nucli, o bé es vol afegir una nova prestació sense generar una versió de nucli nova. Per a corregir el problema (o afegir la nova prestació), se sol distribuir un pedaç en lloc d'un nou nucli complet. Quan ja hi ha uns quants pedaços, s'uneixen amb diverses millores del nucli anterior per a formar-ne una nova versió. En tot cas, si tenim maquinari problemàtic o l'error afecta la funcionalitat o l'estabilitat del sistema i no podem esperar a la següent versió del nucli, serà necessari aplicar els pedaços.

El pedaç s'acostuma a distribuir en un fitxer comprimit tipus bz2 (bunzip2, encara que també pot trobar-se en gzip amb extensió `.gz`, o xz amb extensió `.xz`), com per exemple podria ser:

```
patchxxxx-version-pversion.xz
```

on `xxxx` sol ser algun missatge sobre el tipus o finalitat del pedaç, `version` seria la versió del nucli a la qual s'aplicarà el pedaç, i `pversion` faria referència a la versió del pedaç, del qual també en poden existir més d'una. Cal tenir en compte que estem parlant d'aplicar pedaços als paquets font del nucli (normalment instal·lats, com vam veure, en `/usr/src/linux` o directori similar de l'usuari utilitzat en la compilació del nucli).

Una vegada disposem del pedaç, haurem d'aplicar-lo. Veurem el procés que cal seguir en algun fitxer *Readme* que acompanya el pedaç, però generalment

el procés segueix els passos (una vegada comprovats els requisits previs) de descomprimir el pedaç en el directori dels fitxers font i aplicar-lo sobre les fonts del nucli, com per exemple:

```
cd /usr/src/linux (o /usr/src/linux-version, o directori que fem servir).
unxz patch-xxxxx-version-pversion.xz
patch -p1 < patch-xxxxx-version-pversion
```

També es pot aplicar prèviament amb l'opció `patch -p1 -dry-run` que només procedeix a fer un primer test, per a assegurar-nos que no hi hagi alguna condició d'error quan se substitueixi el codi. Si no hi ha error, tornem a aplicar-lo llavors sense l'opció de test.

Posteriorment, una vegada aplicat el pedaç, haurem de recompilar el nucli per a tornar-lo a generar.

Els pedaços es poden obtenir de diferents llocs. El més normal és trobar-los en el lloc de magatzem dels nuclis *vanilla* (<http://www.kernel.org>), que en té un arxiu complet. Determinades comunitats Linux (o usuaris individuals) també solen oferir algunes correccions, però és millor buscar en els llocs estàndard per a assegurar un mínim de confiança en aquests pedaços i evitar problemes de seguretat amb possibles pedaços “pirata”. Una altra via és el fabricant de maquinari que pot oferir certes modificacions del nucli (o de controladors en forma de mòduls dinàmics de nucli) perquè funcionin millor els seus dispositius (un exemple conegut és NVIDIA i els seus controladors Linux propietaris per a les seves targetes gràfiques).

Finalment, assenyalarem que moltes distribucions de GNU/Linux (Fedora / Red Hat, Debian) ja ofereixen nuclis pedaç per ells mateixos i sistemes per a actualitzar-los (alguns fins i tot de manera automàtica, com en el cas de Fedora/Red Hat i Debian). Normalment, en sistemes de producció és més recomanable seguir les actualitzacions del fabricant, encara que aquest no oferirà necessàriament l'últim nucli publicat, sinó el que creï més estable per a la seva distribució, amb l'inconvenient de perdre prestacions d'última generació o alguna novetat en les tècniques incloses en el nucli.

Finalment, cal comentar la incorporació d'una tecnologia comercial a l'ús de pedaços en Linux, Ksplice (mantinguda per Oracle), que permet a un sistema Linux afegir pedaços al nucli sense necessitat d'aturar i tornar a arrancar el sistema. Bàsicament, Ksplice determina a partir dels paquets font quins són els canvis introduïts per un pedaç o una sèrie de pedaços, i comprova en memòria com afecten la imatge del nucli en memòria que s'està executant. S'intenta llavors aturar l'execució en el moment en què no hi hagi dependències de tasques que necessitin les parts del nucli que cal apedaçar. Llavors es procedeix a canviar, en el codi objecte del nucli, les funcions afectades, apuntant les noves

Nucli actualitzat

En sistemes que es vulguin tenir actualitzats, per raons de test o de necessitat de les últimes prestacions, sempre es pot acudir a <http://www.kernel.org> i obtenir el nucli més modern publicat, quan la compatibilitat de la distribució ho permeti.

Ksplice

Ksplice és una tecnologia molt útil per a servidors empresarials en producció. Podeu consultar-ne el web: <http://www.ksplice.com>

funcions amb el pedaç aplicat i modificant dades i estructures de memòria que hagin de reflectir els canvis. Actualment és un producte comercial (d'Oracle), però certes distribucions de comunitat ja l'estan incloent a causa del suport gratuït que algunes ofereixen (entre aquestes, Ubuntu Desktop i Fedora). En els casos de producció a empreses, amb servidors en els quals és important no disminuir el temps de servei, pot ser una tecnologia crítica per a fer servir, altament recomanable tant per a disminuir el temps de pèrdua de servei com per a minimitzar incidents de seguretat que afectin el nucli.

Bàsicament, ofereixen un servei denominat *Ksplice Uptrack* que és una espècie d'actualitzador de pedaços per al nucli en execució. La gent de Ksplice segueix el desenvolupament dels pedaços font del nucli, els prepara en forma de paquets que es puguin incorporar a un nucli en execució i els fa disponibles en aquest servei *uptrack*. Una eina gràfica gestiona aquests paquets i els fa disponibles per a l'actualització durant l'execució.

SUSE també va anunciar la seva idea d'integrar una tecnologia semblant, denominada *kGraft*, com a mecanisme per a aplicar pedaços al kernel sense reiniciar el sistema, de manera semblant a *Ksplice*. Tanmateix, en aquest cas la idea dels desenvolupadors és intentar incloure-la en la línia principal del kernel, de manera que estaria disponible com a part del codi del kernel. En aquesta tecnologia, de la qual ja hi ha algunes versions que s'estan integrant en les versions de desenvolupament del kernel, bàsicament el pedaç s'integra com un mòdul del kernel (.ko), el qual és carregat amb *insmod*, i reemplaça les funcions kernel afectades pel pedaç, fins i tot durant el temps d'execució d'aquestes funcions. La idea de kGraft és permetre arreglar *bugs* crítics de kernel sense afectar els sistemes que necessiten gran estabilitat i disponibilitat.

kGraft

Nova tecnologia de SUSE per a pedaç en execució del kernel:
<https://www.suse.com/promo/kgraft.html>

5. Mòduls del nucli

El nucli és capaç de carregar dinàmicament porcions de codi (mòduls) a demanda [Hen] per a complementar la seva funcionalitat (es disposa d'aquesta possibilitat des de la versió 1.2 del nucli). Per exemple, els mòduls poden afegir suport per a un sistema de fitxers o per a dispositius de maquinari específics. Quan la funcionalitat proporcionada pel mòdul no és necessària, el mòdul pot ser descarregat i així alliberar memòria.

Flexibilitat del sistema

Els mòduls aporten una flexibilitat important al sistema, i permeten que s'adapti a situacions dinàmiques.

Normalment, a demanda, el nucli identifica una característica no present en el nucli en aquell moment, contacta amb un fil (*thread*) del nucli anomenat `kmod` (en les versions del nucli 2.0.x el dimoni es deia `kerneld`) i aquest executa una ordre `modprobe` per intentar carregar el mòdul associat a partir d'una cadena amb el nom de mòdul o bé d'un identificador genèric. Aquesta informació en forma d'àlies entre el nom i l'identificador es consulta en el fitxer `/etc/modules.conf`, encara que en les recents distribucions s'ha migrat a una estructura de subdirectoris on s'inclou un fitxer d'opcions per mòdul, que es pot veure en el subdirectori `/etc/modprobe.d`.

A continuació, es busca a `/lib/modules/version-kernel/modules.dep` per a saber si hi ha dependències amb altres mòduls. Finalment, amb l'ordre `insmod` es carrega el mòdul des de `/lib/modules/version_kernel/` (el directori estàndard per als mòduls), la `version-kernel` és la versió del nucli actual i s'utilitza l'ordre `uname -r` per a determinar-la. Per tant, els mòduls en forma binària estan relacionats amb una versió concreta del nucli, i solen col·locar-se a `/lib/modules/version-kernel`. Els mòduls es reconeixen com a arxius dins de l'estructura d'aquest directori, amb `.ko` com a extensió d'arxiu.

En general, l'administrador ha de saber com es carreguen els mòduls en el sistema. La majoria de vegades, per mitjà del procés anterior, els mòduls de gran part del maquinari i necessitats concretes són detectats automàticament en l'arrencada o per demanda d'ús i carregats en el moment corresponent. En molts casos no haurem de dur a terme cap procés com a administradors. Però en alguns casos, caldrà preparar alguna sintonització del procés o dels paràmetres dels mòduls o, en alguns altres, afegir nous mòduls ja en forma binària o per compilació a partir dels fitxers font.

Si cal compilar alguns mòduls a partir de les seves fonts, s'ha de disposar dels paquets font i/o *headers* de la versió del nucli al qual està destinat.

Hi ha unes quantes utilitats que ens permeten treballar amb mòduls (solien aparèixer en un paquet de programari anomenat `modutils`, que es va reemplaçar per `module-init-tools`):

- `lsmod`: podem veure els mòduls carregats en el nucli (la informació s'obté del pseudofitxer `/proc/modules`). Es crea la llista dels noms, de les dependències amb d'altres (entre claudàtors []), de la mida del mòdul en bytes i del comptador d'ús del mòdul; això permet descarregar-lo si el recompte és zero.

Exemple

Alguns mòduls en un Debian:

Module	Size	Used by	Tainted: P
agpgart	37344	3	(autoclean)
apm	10024	1	(autoclean)
parport_pc	23304	1	(autoclean)
lp	6816	0	(autoclean)
parport	25992	1	(autoclean) [parport_pc lp]
snd	30884	0	
af_packet	13448	1	(autoclean)
nvidia	1539872	10	
es1371	27116	1	
soundcore	3972	4	[snd es1371]
ac97_codec	10964	0	[es1371]
gameport	1676	0	[es1371]
3c59x	26960	1	

- `modprobe`: intenta la càrrega manual d'un mòdul i de les seves dependències.
- `insmod`: carrega un mòdul determinat.
- `depmod`: analitza dependències entre mòduls i crea un fitxer de dependències.
- `rmmmod`: treu un mòdul del nucli.
- `depmod`: es fa servir per a generar el fitxer de dependències dels mòduls, que es troba a `/lib/modules/version-kernel/modules.dep` i que inclou les dependències de tots els mòduls del sistema. Si s'instal·len nous mòduls de nucli, és interessant executar manualment aquesta ordre per a actualitzar les dependències. També se solen generar automàticament en engegar el sistema.
- Es poden utilitzar altres ordres per a depurar o analitzar els mòduls, com per exemple `modinfo`, que crea una llista d'algunes informacions associades al mòdul (com ara llicències, descripció, ús i dependències), o de fitxers `/proc/kallsyms`, que ens permeten examinar els símbols exportats pels mòduls.

Normalment per a la càrrega, o bé el mateix nucli o bé l'usuari, a mà, especificarà amb `insmod` el nom del mòdul i, opcionalment, determinats paràmetres; per exemple, en el cas de dispositius és habitual especificar les adreces dels ports d'E/S o bé els recursos d'IRQ o DMA. Per exemple:

```
insmod soundx io=0x320 irq=5
```

La càrrega general de mòduls, en funció del moment i la manera, pot fer-se manualment, com hem comentat, mitjançant `initrd/initramfs` (que implica una precàrrega en temps d'arrencada del sistema) o mitjançant `udev` (en situacions dinàmiques d'ús de dispositius removibles, o de connexió en calent).

En el cas de `initrd/initramfs`, quan el sistema arrenca, es necessiten immediatament alguns mòduls per a accedir al dispositiu i al sistema de fitxers arrel del sistema (per exemple, controladors específics de disc o tipus de sistemes de fitxers). Aquests mòduls necessaris es carreguen a la RAM mitjançant un sistema de fitxers especial anomenat `initrd/initramfs`. En funció de la distribució GNU/Linux, s'utilitzen aquests termes de manera indiferent, tot i que en alguns casos s'han produït canvis al llarg de la vida de la distribució. Per convenció, aquest element s'acostuma a anomenar *filesystem* RAM inicial, i se'l referencia com a `initramfs`.

El sistema inicial de fitxers en RAM, `initramfs`, és carregat pel *bootloader* en l'especificació de l'entrada pertinent a la càrrega del nucli corresponent (per exemple en la línia/opció `initrd` de l'entrada corresponent de Grub).

En la majoria de distribucions, amb el nucli distribuït originalment o bé amb la nostra configuració del nucli, acostuma a crear-se un `initramfs` inicial. En qualsevol cas, si depenent de la distribució no es produeix (pot no ser necessari), podem crear-lo i sintonitzar-lo manualment. L'ordre `mkinitramfs` (o una posterior ordre *update-initramfs*) permet crear-lo a partir de les seves opcions genèriques, que es poden configurar en l'arxiu

```
/etc/initramfs-tools/initramfs.conf
```

i, de manera específica, els mòduls que es carregaran en inici automàticament, i que podem trobar-los en `/etc/initramfs-tools/modules`.

Un `mkinitramfs -o new_initrd_file` ens permetrà crear-lo, i normalment podem procedir a copiar-lo en el directori `/boot` per a fer-lo accessible al *bootloader* utilitzat. Per exemple, mitjançant un canvi en el fitxer de configuració `/boot/grub/menu.lst` de Grub, amb la modificació de la línia de `initrd` oportuna. En qualsevol cas, sempre és interessant establir al *bootloader* una configuració alternativa durant aquestes proves, per a poder reiniciar i provar la nova configuració però, alhora, mantenir la configuració estable antiga.

Durant el procés d'arrencada, a més del `initramfs` necessari per a l'arrencada inicial, es carregará la resta de mòduls per detecció automàtica. Si no es carrega algun mòdul, sempre pot forçar-se'n la càrrega en incloure'n el nom implícitament en el fitxer de configuració `/etc/modules`.

També es pot donar o voler el cas contrari: evitar la càrrega d'un mòdul que es pot detectar erròniament o per al qual existeix més d'una alternativa. En aquest cas s'utilitzen tècniques de llistes negres de mòduls (típicament la llista negra es desa a `/etc/modprobe.d/blacklist.conf`), tot i que es pot crear en un fitxer arbitrari en aquest directori, simplement afegint una línia *blacklist nommodul*.

5.1. DKMS: mòduls recompilats dinàmicament

Pel que fa als mòduls dinàmics, un problema clàssic ha estat la recompilació amb noves versions del nucli. Els mòduls dinàmics de tercers, no inclosos *a priori* en el nucli, necessiten codi font per a compilar-se, proporcionat per la comunitat o pel fabricant del maquinari del dispositiu.

Durant la compilació, normalment cal disposar dels paquets de desenvolupament del nucli, dels paquets del codi font del nucli i dels seus *headers* de desenvolupament, amb la mateixa numeració que el nucli utilitzat per al qual es vol compilar el mòdul. Aquest fet obliga a una recompilació constant amb el canvi de versions del nucli del sistema, en especial ara que les distribucions distribueixen revisions del nucli en un temps més breu, o bé per a resoldre problemes potencials de seguretat o bé per a corregir errors detectats.

Algunes distribucions, per a minimitzar aquesta problemàtica, distribueixen un entorn anomenat DKMS, que permet facilitar la recompilació automàtica d'un mòdul amb els canvis de versió del nucli. Això normalment es produeix en l'arrencada en detectar un número de nucli: tots els mòduls de tercers registrats pel sistema DKMS es recompilen a partir dels arxius font dels mòduls i dels arxius font o *headers* del nucli nou. D'aquesta manera, el procés total és transparent a l'usuari. Una vegada fet aquest procés, el sistema o l'usuari, mitjançant ordres de manipulació de mòduls (com `modprobe`), poden utilitzar directament el nou mòdul, o si estava configurat en arrencada, un cop produïda aquesta, el nou mòdul s'utilitzarà.

Algunes distribucions ofereixen només el paquet base (`dkms`) del sistema, en algunes es proporcionen paquets `dkms` preparats per a mòduls concrets o, fins i tot, el fabricant pot oferir el seu mòdul amb suport `dkms`.

El procés habitualment passa per les etapes següents:

- 1) Obtenir els arxius font del mòdul (un paquet o un arxiu TGZ acostuma a ser el més normal).
- 2) Instal·lar els diversos paquets necessaris per al procés: `kernel-source`, `kernel-headers`, `dkms` (els noms depenen de la distribució i, en el cas dels paquets font del nucli, cal tenir en compte que siguin les versions associades a la versió del nucli actual per al qual es vol instal·lar el mòdul); normalment n'hi ha prou amb els *headers*, però depenent del mòdul poden ser necessaris més paquets de fonts.
- 3) Activar el servei DKMS en arrencada. Habitualment, el servei s'anomena `dkms_autoinstaller`.

- 4) Crear un directori a `/usr/src` per als paquets font del mòdul i col·locar-los-hi.
- 5) Crear un fitxer `dkms.conf` en el directori anterior, que especifica com construir (compilar) i instal·lar el mòdul.
- 6) Afegir el servei a la base de dades DKMS, compilar-lo i instal·lar-lo, normalment amb unes ordres:

```
dkms add -m nom-modul -v numero-versio-modul
dkms build -m nom-modul -v numero-versio-modul
dkms install -m nom-modul -v numero-versio-modul
```

Respecte al fitxer `dkms.conf` esmentat, podria ser com segueix (en aquest cas `nom-modul` és el nom del mòdul i `versio`, el codi numèric de la versió):

```
#
# /usr/src/nom-modul/dkms.conf
#

PACKAGE_NAME="nom-modul"
PACKAGE_VERSION="versio"
CLEAN="make clean"
MAKE[0]="make module"
BUILD_MODULE_NAME[0]="nom-modul"
DEST_MODULE_LOCATION[0]="/kernel/drivers/video"
AUTOINSTALL="yes"

# End Of File
```

En aquest cas, tenim localitzats els paquets font del mòdul en un directori `/usr/src/nom-modul` en què posem aquest fitxer `dkms.conf`. L'opció `MAKE` dóna els passos per a compilar i construir el mòdul, prèviament netejat amb `CLEAN`. `BUILD_MODULE_NAME` estableix el nom del mòdul construït (cal anar amb compte en aquest punt perquè depèn del sistema de compilació i pot coincidir amb el nom del mòdul general o no; alguns paquets font permeten construir diversos controladors/mòduls diferents, amb diferent denominació).

`DEST_MODULE_LOCATION` defineix on s'instal·larà el mòdul final en l'arbre associat al nucli; en aquest cas suposem que és un controlador de vídeo (recordeu que l'arrel és a `/lib/modules/version-kernel`, el que es col·loca aquí és a partir d'aquesta arrel). `AUTOINSTALL` permet que es reconstrueixi automàticament el mòdul durant canvis del nucli actual.

En els casos de

`CLEAN`, `MAKE`, `BUILD_MODULE_NAME` i `DEST_MODULE_LOCATION`

es recomana consultar el fitxer d'explicació (normalment amb nom `README` o `INSTALL`) que acompanya els paquets font dels mòduls, ja que poden caldre ordres addicionals, o es poden haver de modificar perquè es compili i s'instal·li correctament el mòdul.

6. Virtualització en el nucli

Una de les àrees en expansió, en l'administració d'IT, és la virtualització de sistemes i la seva integració amb entorns *cloud* de tipus públic/privat, com veurem més endavant. Amb el temps, GNU/Linux ha anat incorporant diferents possibilitats provinents tant de solucions comercials com de diferents projectes de codi obert.

La virtualització de sistemes és un recurs bàsic actual en les empreses i organitzacions per a millorar-ne l'administració de sistemes, disminuir costos i aprofitar els recursos de maquinari de manera més eficient.

En general, quan en el passat necessitàvem diverses instàncies d'un o més sistemes operatius, havíem d'adquirir un servidor per a cada instància que volguéssim implantar. El corrent actual és comprar servidors molt més potents i utilitzar la virtualització en aquests servidors per a implantar els diferents sistemes en desenvolupament o producció.

Normalment, en virtualització disposem d'un sistema operatiu instal·lat (que habitualment s'anomena *sistema amfitrió* o *host*) i d'una sèrie de màquines virtuals sobre aquest sistema (anomenades *sistemes hoste* o *guest*). Tot i així, també hi ha solucions que substitueixen el sistema operatiu amfitrió per una capa anomenada *hypervisor*.

La virtualització com a solució ens permet optimitzar l'ús dels nostres servidors o, per exemple en el cas d'escriptori, disposar de màquines de prova d'altres sistemes operatius convivint alhora en la mateixa màquina. En el cas de GNU/Linux disposem de múltiples solucions que permeten tant un cas com l'altre. També podem disposar de GNU/Linux com a sistema amfitrió que allotja màquines virtuals o bé utilitzar-lo com a màquina virtual sobre un altre sistema diferent o bé sobre un altre sistema amfitrió també GNU/Linux. Un esquema, aquest darrer, particularment útil en el cas d'administració, perquè ens permetrà, per exemple, examinar i executar diferents distribucions GNU/Linux sobre un mateix sistema amfitrió base, per a observar diferències entre aquestes o personalitzar les nostres tasques o *scripts* per a cada distribució.

Hi ha moltes solucions de virtualització, però per esmentar-ne algunes de les més populars en sistemes GNU/Linux, disposem de les següents (les ordenem de més a menys en relació directa amb el nucli):

- KVM
- Xen

- LXC
- OpenVZ
- VirtualBox
- VMware

VMware és un dels líders comercials en solucions de virtualització, i disposa de productes de virtualització per a escriptori (VMware Workstation / Fusion), mentre que per a servidor disposa d'un producte VMware vSphere que està disponible per a Linux i es pot descarregar gratuïtament. El cas de servidor permet gestionar diverses màquines virtuals amb una interfície de gestió simple. Altres línies de productes VMware implementen necessitats més grans per a centres de dades (*data centers*) i entorns de *cloud computing*, amb gestió elaborada de màquines, tolerància a errors, migracions i altres necessitats explícites per a centres de dades.

Oracle VirtualBox ofereix virtualització orientada a escriptori, que ens permet una opció bastant senzilla per a provar màquines amb diferents sistemes operatius. Disposa de versió de codi lliure utilitzable en gran quantitat de distribucions GNU/Linux.

OpenVZ és una solució de virtualització que utilitza el concepte de *contenedor de màquina virtual*. Així, l'amfitrió arrenca amb un nucli comú a les màquines virtuals (hi ha paquets d'imatges de nucli amb suport OpenVZ integrat en les distribucions, com per exemple en Debian), que permet arrencar màquines amb el nucli en comú però cada una dins d'un entorn aïllat de la resta. OpenVZ només permet màquines virtuals hoste Linux (a causa del nucli compartit).

LXC (Linux Containers) és un sistema de virtualització a escala operativa que permet córrer múltiples sistemes Linux de manera aïllada. En lloc de crear màquines virtuals completes, es basa en l'ús d'una funcionalitat del kernel de Linux, denominada *cgroups*, juntament amb *chroot*, que permet encapsular un espai d'execució de processos i gestió de xarxa de manera aïllada. La idea és similar a OpenVZ, però en aquest cas no necessita actuacions de mòduls o pedaços de kernel, ja que funciona sota el kernel *vanilla*, sense modificacions. Alguns sistemes per desplegament de contenidors, com *Docker*, l'utilitzen com a base per a implementar els contenidors.

Xen usa el concepte d'*hypervisor*, utilitzant un tipus de virtualització denominada *paravirtualització*, en què s'elimina el concepte d'amfitrió hoste (*host-guest*) i es delega a la capa d'*hypervisor* la gestió dels recursos físics de la màquina, de manera que permeti a les màquines virtuals el màxim accés als recursos de maquinari. En aquests casos es necessiten nuclis sintonitzats que puguin beneficiar-se de les possibilitats de la paravirtualització. En el cas de GNU/Linux, en la majoria de distribucions s'ofereixen nuclis optimitzats per a Xen (vegeu el cas de Debian, per al qual existeixen imatges binàries dels

Enllaç d'interès

Podeu visitar el web de VMware a:
<http://www.vmware.com>

nuclis per a Xen). En general, la paravirtualització i la capa d'*hypervisor* per a accedir al maquinari aprofiten les facilitats de les CPU actuals, amb recursos de maquinari dedicats a facilitar la virtualització. Si es disposa de les característiques, es poden usar sistemes com Xen, si no, es pot utilitzar un sistema més clàssic de d'amfitrió hoste, com per exemple VirtualBox.

En general, per a veure si la CPU té suport de virtualització, cal examinar-ne dades a `/proc/cpuinfo`, en concret el *flag* `VMX`, per a processadors Intel, o `svm` en processadors AMD, que es pot veure en la secció `flags`:

```
$ cat /proc/cpuinfo
processor       : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 23
model name    : Intel(R) Xeon(R) CPU E5405  @ 2.00GHz
stepping      : 10
cpu MHz       : 1994.999
cache size    : 6144 KB
physical id   : 0
siblings      : 4
core id       : 0
cpu cores     : 4
apicid        : 0
fpu           : yes
fpu_exception : yes
cpuid level   : 13
wp            : yes
flags         : fpu vme de pse tsc msr pae mce cx8 apic sep
mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2
ss ht tm syscall nx lm constant_tsc pni monitor ds_cpl
vmx tm2 ssse3 cx16 xtpr sse4_1 lahf_lm
bogomips      : 3989.99
clflush size  : 64
cache_alignment : 64
address sizes  : 38 bits physical, 48 bits virtual
power management:
```

6.1. KVM

La solució en la qual ens centrarem en aquest subapartat, **KVM**, és present des del nucli 2.6.20, com a solució inclosa per a la virtualització de sistemes. És una solució semblant a Xen, però amb diferències quant a la implementació. Xen és un producte complex, amb diferents capes i, en especial, el seu disseny de capa hipervisora. En canvi, KVM s'implementa com un mòdul del nucli

Enllaç d'interès

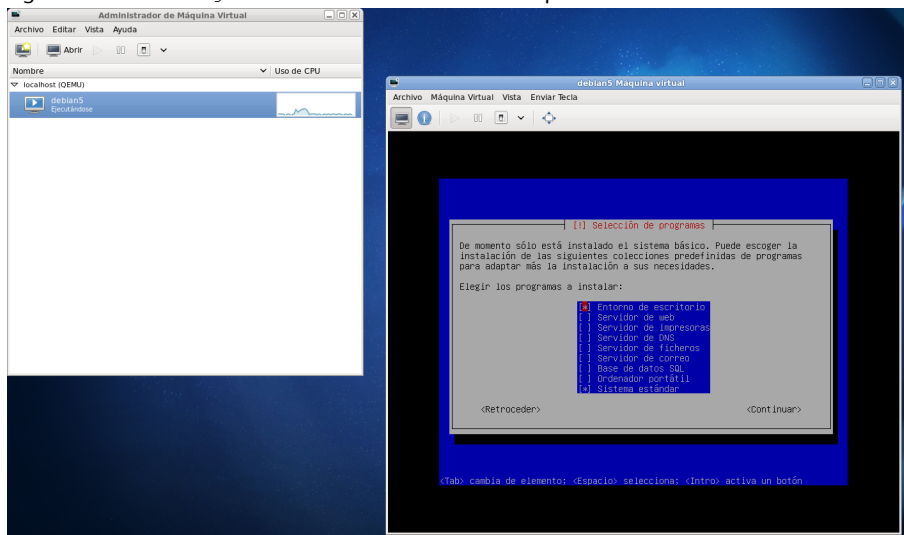
Enllaç web de KVM a:
<http://www.linux-kvm.org>

existent, que es complementa amb altres solucions. És l'opció per defecte per a la virtualització en distribucions com Debian i Fedora.

Normalment, una solució de virtualització basada en KVM es compon d'una barreja del següent:

- El mòdul de nucli `kvm.ko`, que proporciona la infraestructura base de virtualització, i, a més, un mòdul addicional segons el model de processador, `kvm-intel.ko` o `kvm-amd.ko`. Aquests mòduls s'hauran de carregar manualment (`modprobe`) o habilitar-los durant l'arrencada per a disposar de suport de virtualització KVM.
- Qemu, un simulador creuat d'arquitectures de CPU que ens permet simular una CPU virtual sobre una altra d'arquitectura real igual o diferent. KVM utilitza una versió modificada de Qemu per a la gestió i la creació de les màquines hoste (aquest paquet sol aparèixer com a `qemu-kvm` en les distribucions).
- La biblioteca `libvirt`, que és una API que permet una gestió de la virtualització independent del sistema de virtualització utilitzat (Xen, KVM o d'altres).
- Utilitats basades en `libvirt` per a la creació de les VM hoste i el seu manteniment, com `virt-manager`, una interfície gràfica de gestió de màquines virtuals (figura 3); `virt-install`, una utilitat de línia per a la gestió de màquines virtuals, o `virtsh`, un intèrpret d'ordres (*shell*) basat en ordres de gestió de les màquines virtuals. Aquestes utilitats solen necessitar un servei de sistema, anomenat `libvirtd` (en algunes distribucions `libvirt-bin`). Hem d'assegurar-nos de posar en marxa el servei (`/etc/init.d/libvirtd start`, `/etc/init.d/libvirt-bin start` o equivalents amb `Systemd`) o bé de carregar-lo a l'inici.

Figura 3. Virt-manager durant la instal·lació d'una màquina virtual



A continuació, veurem els processos bàsics associats a la utilització de KVM i descriurem algunes de les fases de la instal·lació de KVM i la posada en marxa d'algunes màquines virtuals.

Per començar, hem d'examinar si disposem de suport de maquinari a la CPU: com hem esmentat, busquem el *flag* `vmx` a `/proc/cpuinfo` (per a processadors Intel) o el *flag* `svm` (per a processadors AMD). Per exemple, amb (substitueix `vmx` per `svm` en AMD):

```
grep vmx /proc/cpuinfo
```

obtindrem com a resultat la línia de *flags* (si existeix el *flag* `vmx`, si no, cap resultat). També podem obtenir diverses línies en CPU *multicore* i/o Intel amb *hyperthreading*, on obtenim una línia de *flags* per cada element de còmput (CPU amb HT o múltiples nuclis amb HT o sense).

Una vegada determinat el suport correcte de virtualització, instal·lem els paquets associats a KVM; aquests ja dependran de la distribució, però en general, el mateix “kvm” ja obtindrà la majoria de paquets requerits com a dependències. En general, els paquets recomanats per a instal·lar (via `apt` o `yum`) són `kvm`, `qemu-kvm`, `libvirt-bin`, `virt-viewer` entre altres. Depenent de la distribució, poden canviar alguns noms de paquets i, en especial, amb el nom `virt` hi ha diversos paquets d'utilitats de generació i monitoratge de màquines virtuals, tant de KVM com de Xen, ja que la biblioteca `libvirt` ens permet abstroure diversos sistemes diferents de virtualització.

Si es permet als usuaris del sistema l'ús de màquines virtuals, cal afegir els noms d'usuari a grups específics (via ordres `adduser` o `usermod`), normalment als grups `kvm` o `libvirt` (depenent de la distribució, Fedora o Debian, i de la versió de KVM):

```
# adduser <usuari> libvirt
```

En aquest moment, podem executar l'ordre de *shell* per a la gestió de màquines KVM, `virsh`, que ens permetria veure les disponibles:

```
$ virsh --connect qemu:///system list
```

això permet connectar-se al gestor local, i preguntar-li per la llista de màquines disponibles.

El pas següent (opcional) és facilitar l'ús de xarxa a les màquines virtuals. Per defecte, KVM utilitza NAT, i dona adreces IP privades de tipus 10.0.2.x, i hi accedeix mitjançant la xarxa de la màquina amfitrió. En un altre cas, si volem

Suport de virtualització

Cal anar amb compte amb el fet que moltes de les màquines actuals permeten desactivar/activar a la BIOS el suport de virtualització; comproveu primer que no estigui desactivat.

una configuració diferent (per exemple que permeti accés extern a les màquines virtuals) haurem de permetre que la xarxa actual faci de *bridge*; en aquest cas és necessari instal·lar el paquet `bridge-utils` i configurar un dispositiu especial de xarxa denominat `br0`: en Debian en la configuració de xarxa que es troba a `/etc/network/interface` i en Fedora pot crear-se un fitxer associat al dispositiu com ara `/etc/sysconfig/network-scripts/ifcfg-br0`. Per exemple, en Debian podria col·locar-se un exemple de configuració com el següent:

```
auto lo
iface lo inet loopback

auto br0
iface br0 inet static
    address 192.168.0.100
    network 192.168.0.0
    netmask 255.255.255.0
    broadcast 192.168.0.255
    gateway 192.168.0.1
    bridge_ports eth0
    bridge_fd 9
    bridge_hello 2
    bridge_maxage 12
    bridge_stp off
```

Aquesta configuració permet que es creï un dispositiu `br0` per a reemplaçar `eth0`. Així, les targetes de xarxa virtuals redirigiran el trànsit assignat a aquest dispositiu. `bridge_ports` especifica quin serà el dispositiu físic real que s'utilitzarà.

Tal com hem comentat, aquesta part de configuració de xarxa és opcional, i només té sentit si volem accedir des de l'exterior a les nostres màquines virtuals. Ben al contrari, en un entorn de virtualització d'escriptori pot haver-n'hi prou amb el mode NAT per defecte, ja que les màquines disposaran de sortida de xarxa a través de la xarxa de l'amfitrió.

A partir d'aquestes configuracions, ja estarem capacitats per a crear les imatges de les màquines virtuals. Hi ha diferents conjunts d'ordres per a fer-ho, fent servir `kvm` directament (ordre `kvm`), utilitats associades a `qemu` per a la creació d'aquestes imatges (`qemu-img`) o utilitats associades a `libvirt` (`virt-install`). El procés passa per crear la imatge de disc (o espai de disc) associat a la màquina `hoste` com un fitxer i, després, fer la instal·lació del sistema operatiu en aquest fitxer (imatge de disc), des del CD/DVD d'instal·lació del sistema operatiu o també des d'una imatge `*.iso` del CD/DVD.

Per exemple, suposem una instal·lació d'un `hoste` determinat (per exemple, disposem d'uns CD/DVD o d'arxius d'imatges ISO de la distribució Debian).

Enllaç d'interès

Per altra banda, un tema important és la gestió de xarxa, que és un dels temes complexos en virtualització; en el cas de KVM es recomana examinar: <http://www.linux-kvm.org/page/Networking>.

Creem l'espai de disc per a la màquina virtual (en aquest cas, 8 GB):

```
# dd if=/dev/zero of=~/.debianVM.img bs=1M count=8192
```

Mitjançant l'ordre `dd`, creem un fitxer de sortida de 8.192 blocs d'1 MB, és a dir, 8 GB, que ens formarà la unitat de disc per a la nostra màquina virtual (també existeix una ordre alternativa per a crear imatges, `qemu-img`, vegeu la pàgina man).

Després, cal obtenir el mitjà d'instal·lació del sistema operatiu a instal·lar, proporcionat com a CD/DVD i, per tant, accessible en el dispositiu `/dev/cdrom` (o equivalent si tenim més unitats de CD/DVD) o, en canvi, a partir d'una imatge `iso` del suport. En qualsevol cas, aquest darrer sempre el podem obtenir a partir dels discos amb

```
dd if=/dev/cdrom of=debian-install.iso
```

que ens genera la imatge del CD/DVD o directament descarregar la imatge ISO de la distribució.

Ara que disposem de la imatge binària i el mitjà d'instal·lació del sistema operatiu, podem instal·lar-lo, amb diferents utilitats. En general, acostuma a utilitzar-se `virt-install` com a ordre per a crear la VM, però també hi ha la possibilitat d'usar el `qemu-kvm` esmentat, directament com a opció més simple, que sol aparèixer (depenent de la distribució) com a ordre `qemu-kvm`, `qemu-system-x86_64` o simplement com a `kvm` (en algunes distribucions, `kvm` ja no es fa servir o només hi és per compatibilitat):

```
$ qemu-system-x86_64 -enable-kvm -m 512 -cdrom debian-install.iso \
-boot d -hda debianVM.img
```

En aquest cas, crearia una màquina virtual (arquitectura `x86_64`) bàsica de 512 MB de memòria principal, fent servir el nostre disc de 8 GB creat prèviament i arrencant la màquina a partir de la nostra imatge `iso` del mitjà d'instal·lació del sistema operatiu. Es pot substituir el fitxer `iso` per `/dev/cdrom` si utilitzem els discos d'instal·lació.

L'altra possible alternativa de creació és mitjançant la utilitat de l'indicador d'ordres `virt-install`, molt més completa, però amb la dificultat d'haver d'especificar molts més paràmetres. Per exemple, podríem indicar la creació de la màquina anterior mitjançant

```
virt-install --connect qemu:///system -n debian -r 512 \
--vcpus=2 -f debianVM.img -s 8 -c debianinstall.iso --vnc \
--noautoconsole --os-type linux --os-variant debianLenny
```

que entre altres paràmetres, col·loca el mètode de connexió a la màquina virtual, el nom de la VM `debian`, defineix 512 MB de memòria, fa disponibles 2 nuclis de la màquina física, utilitza el disc virtual `debianVM`, de 8 GB (–s permet que si no existeix, el creï de manera prèvia amb aquesta grandària de GB), utilitza la `iso` com a mitjà d'instal·lació i permetrà connexions gràfiques amb `vnc` amb la màquina virtual. A més, definim que el sistema que s'instal·larà és Linux, especialment una versió de Debian. Els paràmetres de tipus i variant del sistema operatiu són altament dependents de la versió de `virt-install`, de manera que val la pena consultar la seva pàgina `man (man virt-install)` i la llista de sistemes operatius compatibles amb la versió KVM del nucli i el conjunt d'utilitats `qemu` i `libvirt`.

En aquest punt només hem creat la màquina, però no hi estem connectats, i n'hem configurat de manera molt bàsica els paràmetres. Amb altres utilitats, per exemple les basades en `libvirt`, com la interfície gràfica `virt-manager`, podem personalitzar més la VM creada, per exemple afegint o traient maquinari virtual al sistema. Amb `virt-manager` podem afegir la connexió a la màquina de la qual hem creat la imatge, per exemple en `localhost`, fet que ens permetrà després tenir-la accessible mitjançant connexió a `qemu:///system`, i també arrencar-la de seguida.

Després podem visualitzar la consola (de text o gràfica) de la màquina acabada de crear mitjançant

```
virt-viewer -c qemu:///system nombreVM
```

si és al mateix sistema `localhost` o, si som en una màquina remota, amb

```
virt-viewer -c qemu+ssh://ip/system nombreVM
```

O utilitzant directament la interfície `virt-manager` (figura 4).

Això ens permet connectar gràficament amb la màquina virtual creada i, per exemple, continuar amb la instal·lació del sistema operatiu (s'haurà iniciat amb l'arrencada anterior amb `virt-install` o `qemu-system-x86_64`). Una vegada instal·lat el sistema, ja podem usar qualsevol sistema, tant si és gràfic (`virt-manager`) com d'indicador d'ordres (`virtsh`), per a gestionar les màquines hoste virtuals i poder arrencar-les i parar-les.

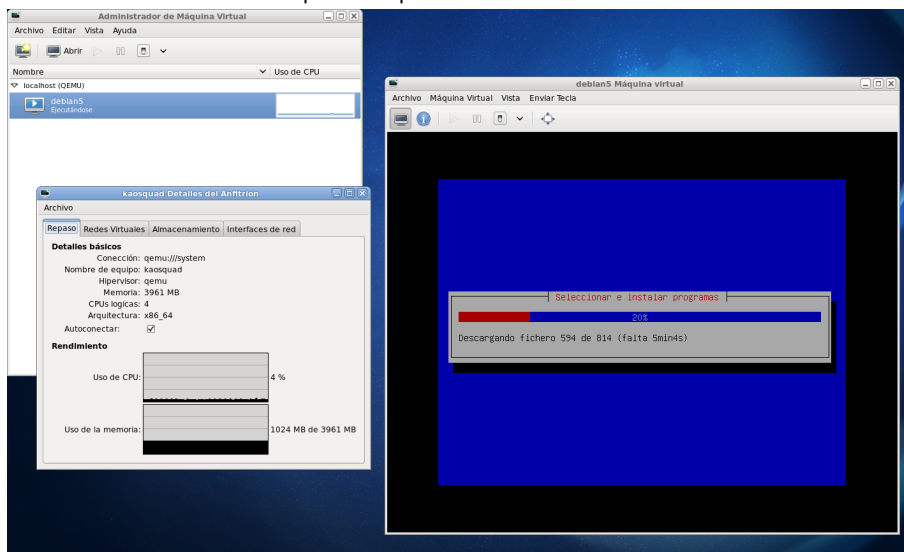
Per exemple, amb `virsh`:

```
virsh --connect qemu:///system
```

Enllaç d'interès

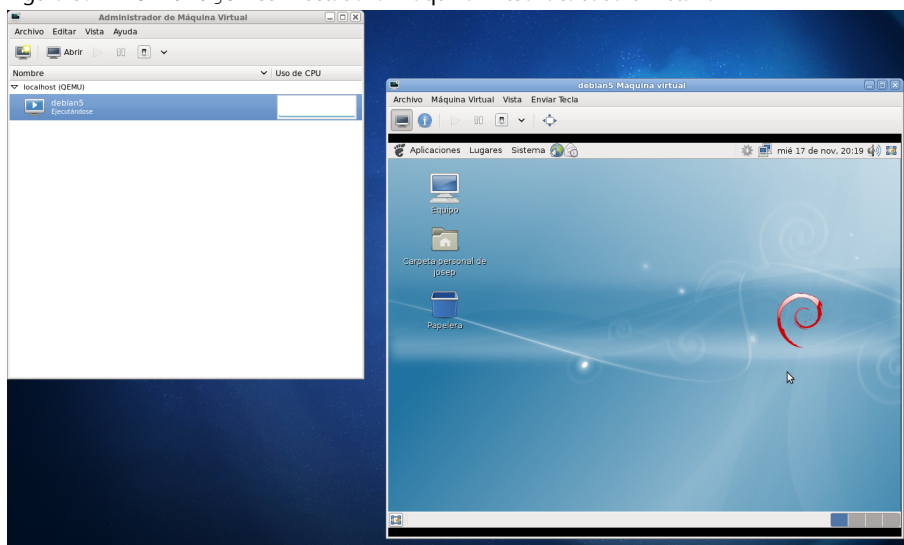
La llista de compatibilitat de KVM pot trobar-se en: http://www.linux-kvm.org/page/guest_support_status

Figura 4. `Virt-manager` connectat a la màquina virtual durant el procés d'instal·lació, observant els recursos utilitzats per la màquina virtual



connectem amb el *shell*, on ordres com `list` ens donen les màquines actives, `list -all`, totes les màquines disponibles, i altres com `start`, `shutdown`, `destroy`, `suspend` o `resume` ens donen diferents possibilitats de gestió de cada màquina virtual.

Figura 5. `Virt-manager` connectat a la màquina virtual acabada d'instal·lar



6.2. VirtualBox

VirtualBox és una solució de virtualització d'escriptori que està obtenint resultats importants, un projecte en aquests moments mantingut per Oracle.

En el cas de Debian, que utilitzarem, està disponible en el repositori oficial. Però per a altres distribucions, poden trobar-se en https://www.virtualbox.org/wiki/linux_downloads paquets preparats per a un gran nombre de distribucions, i repositoris propis que cal afegir a les distribucions.

Aquesta solució de virtualització inclou tant eines de tipus gràfic com utilitats en el nivell de línia d'ordres per a virtualitzar màquines de 32 bits i 64 bits d'arquitectura Intel (x86 i x86_64). VirtualBox s'ofereix majoritàriament amb llicència GPL; tot i així hi ha una part addicional propietària, denominada *Extension Pack*, que ofereix lliure de cost Oracle per a ús personal. Aquest paquet addicional ofereix algunes facilitats extra relacionades amb emulació de maquinari extra (USB2, per exemple, per defecte només s'ofereix USB1 sense pack), i ofereix accés gràfic a les màquines hoste per mitjà d'RDP (*remote desktop protocol*).

Per a la instal·lació en Debian, s'ha d'instal·lar el paquet *VirtualBox* i és recomanat disposar dels *headers* corresponents al kernel actual (paquet *linux-headers-version* i la versió és la corresponent al kernel actual). Els *headers* s'utilitzaran per a la compilació d'una sèrie de mòduls que VirtualBox instal·la dinàmicament en el kernel. Normalment, la instal·lació proporciona també una configuració de *dkms* per a aquests mòduls que permet que es recompilin després de canvis de kernel en la distribució. En algunes distribucions, de vegades passa un temps fins que estan disponibles els mòduls actualitzats per a una versió concreta de kernel; per tant, abans d'actualitzar el kernel en un sistema amb virtualització VirtualBox, és recomanable comprovar que la versió de kernel ja té el suport adequat.

```
# apt-get install linux-headers-version virtualbox
```

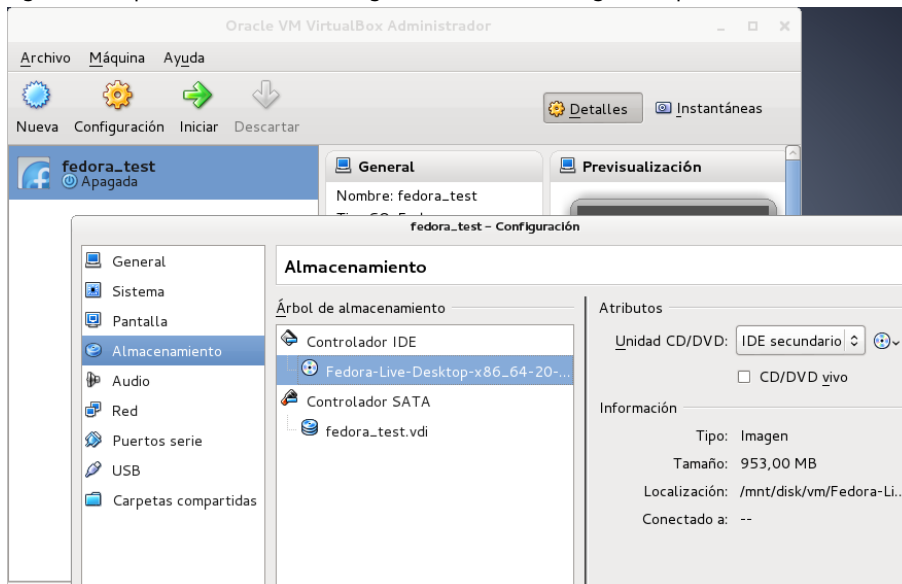
VirtualBox es pot arrencar ja mitjançant l'ordre `virtualbox`. Els mòduls, si teníem els *headers* adequats, s'hauran compilat i carregat en el sistema, i es carregaran en cada arrencada. Si no desitgem que s'arrenquin automàticament, llavors podem canviar en `/etc/default/virtualbox` el paràmetre `LOAD_VBOXDRV_MODULE` i posar-lo a 0.

Un cop arrencat l'entorn gràfic amb VirtualBox, podem crear una nova VM mitjançant l'opció de la icona "Nova", que ens arrencarà l'assistent de creació de màquines virtuals. El procés passa per determinar:

- L'operatiu de la màquina hoste i el seu nom.
- La memòria del nostre sistema, que proporcionarem a la màquina hoste; generalment, es recomana que el total de màquines VM no consumeixin més del 50% de la memòria. Depenent de l'ús, és freqüent de 512 MB a 2 GB per màquina virtual.
- Creació del disc dur virtual de la màquina. Podem crear un disc nou o fer servir una imatge de disc prèviament creada per a la nova màquina.
- Format de la imatge del disc. En VirtualBox, per defecte, és VDI. Tanmateix, podem crear-ne d'altres, que potser puguem compartir amb altres sistemes de virtualització.

- Assignem la grandària del disc virtual de manera dinàmica o reservem el total de l'espai. Això ens permetrà reservar espai de disc a mesura que es vagi fent servir o, per contra, tenir un disc més ràpid, la qual cosa ens permetrà millors prestacions del disc virtual.

Figura 6. Màquina virtual creada en el gestor de VirtualBox afegint ISO per a instal·lar

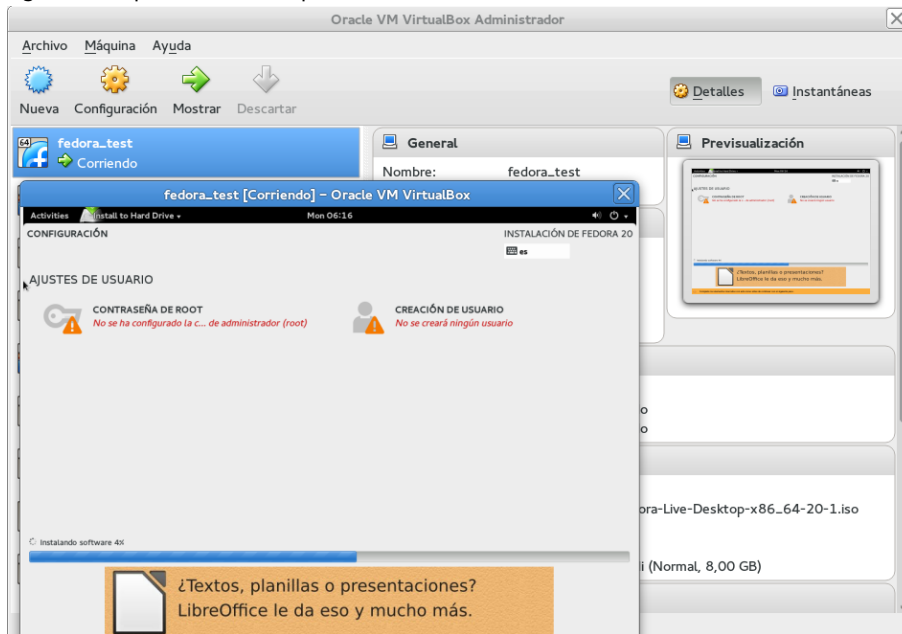


Amb aquest procés, ja disposem d'una màquina creada i podem retocar la seva configuració seleccionant-la; especialment, podem connectar en emmagatzematge una imatge ISO d'una distribució al seu CD-ROM virtual. Això ens permetrà arrencar la màquina des de CD, arrencant així la imatge d'instal·lació. En el procés inicial, també podem canviar la configuració de xarxa; hi ha diverses maneres disponibles: des del NAT, per defecte, a maneres que permeten xarxes privades per a les màquines VM, o incloure una IP pública per a la nostra màquina VM (vegeu documentació VirtualBox per a l'administració de xarxa <http://www.virtualbox.org/manual/usermanual.html>). Vegeu en la figura 6 una màquina creada, amb la seva secció de configuració d'emmagatzematge a la qual s'acaba de connectar una imatge ISO (d'una distribució Fedora), si procedim a l'arrencada de la màquina amb *Iniciar*. En la figura 7 podem veure la màquina hoste durant el procés d'instal·lació.

Una vegada disposem de la VM amb sistema operatiu instal·lat, ja la tindrem funcional. Per a millorar el rendiment de les màquines hoste, VirtualBox també proporciona una sèrie de controladors per al sistema hoste, coneguts com a *VirtualBox Guest Additions*, que permeten millores de la gestió de la xarxa virtual, de l'acceleració dels gràfics i la possibilitat de compartir carpetes amb la màquina *host* (amfitrió). Una vegada arrencada i instal·lada la VM, des del menú Dispositius -> Inserir imatge CD Guest Additions, es poden instal·lar.

Cal assenyalar que, a més de tot aquest procés, determinats proveïdors proporcionen imatges VDI de màquines VirtualBox ja preparades per al seu funcionament, perquè no hàgim de passar per tot el procés de creació. Normalment

Figura 7. Màquina virtual en el procés d'instal·lació d'una distribució GNU/Linux



són imatges denominades *cloud*, i haurem de buscar imatges VDI ja preparades o altres formats compatibles amb VirtualBox. No obstant això, es pot incórrer en certs riscos de seguretat. Convé evitar l'ús de VM ja personalitzades si el proveïdor no és de confiança o no és directament la mateixa distribuïdora; a més, algunes màquines VM arbitràries podrien contenir problemes de seguretat o introduir *malware* o eines destinades a l'atac dels nostres sistemes. Per això, es recomana fer servir només imatges de proveïdors adequats.

Per acabar, cal assenyalar el tema del control mitjançant interfície d'usuari. VirtualBox proporciona les ordres `vboxmanage` i `VBoxHeadless` per permetre'ns la major part de gestió de les màquines des de línia d'ordres, i fins i tot si ja disposem d'imatges VDI, podrem fer tota la gestió sense cap ús d'interfície gràfica.

Un resum d'algunes ordres interessants:

- Arrencar la màquina en mode servidor sense interfície VirtualBox; en arrencar, ens donarà el port de connexió al servidor gràfic (VRDE). També amb `vboxmanage`, amb interfície GUI i sense interfície:

```
$ VBoxHeadless -startvm "nom_maquina"
$ vboxmanage startvm "nom_maquina"
$ vboxmanage startvm "nom_maquina" --type headless
```

- Informació de la màquina virtual, els detalls de la seva configuració, discos, xarxa, memòria, CPU i altres:

```
$ vboxmanage showvminfo "nom_maquina"
```

- Apagar la màquina:

```
$ vboxmanage controlvm "nom_maquina" poweroff
```

- Modificar el servei de pantalla:

```
$ vboxmanage modifyvm "nom_maquina" --vrde on --vrdeport  
port_VRDE --vrdeaddress ip_maquina
```

- Connexió gràfica a la màquina executant-se. Si la màquina està en NAT sense IP assignada, la ip=0.0.0.0 per a la connexió, si no la IP corresponent. Per a ple funcionament es necessiten instal·lades les Guest Additions, les quals tenen com a requisit el paquet de *headers* del kernel corresponent a la màquina hoste:

```
$ rdesktop -a 16 -N ip_maquina:port_VRDE
```

- En particular, amb el següent veurem múltiples opcions de control de les VM que tenim disponibles:

```
$ vboxmanage controlvm
```

6.3. Xen

En el cas de Xen, tractem amb un entorn d'hypervisor en un nivell de màquina física que permet córrer múltiples instàncies del sistema operatiu o de diferents operatius en paral·lel en una sola màquina.

Algunes de les característiques de Xen que cal destacar són les següents:

- L'hypervisor, o component monitor de màquines virtuals, és relativament petit i consumeix pocs recursos de la màquina.
- Xen és agnòstic del sistema operatiu. La majoria d'instal·lacions usen Linux com a sistema base; Xen aprofita els serveis bàsics, el denominat *domini 0* de Xen. Dom0 és una VM especialitzada amb privilegis per a l'accés directe al maquinari. Com a base per a Xen, es poden fer servir altres operatius, com ara alguns de la família BSD.
- Xen és capaç d'aïllar el funcionament d'un *driver* de sistema en una màquina virtual. Si aquest falla o és compromès, la VM que conté el *driver* es pot tornar a arrancar sense afectar la resta del sistema.
- És possible distingir dos modes de funcionament. Un és el denominat *Xen Full Virtualization* (HVM), que fent servir extensions maquinari de virtualització (les HVM), utilitza emulació de PC (Xen es basa en Qemu per a això);

Enllaç d'interès

Es recomana llegir la introducció http://wiki.xen.org/wiki/Xen_Overview, per a comprendre alguns conceptes de virtualització associats a Xen.

en aquest cas no es requereix suport de kernel, però per contra acostuma a ser una solució de menor rendiment a causa de l'emulació necessària. Per altra banda, el mode de funcionament *Xen Paravirtualization* (PV) és una virtualització eficient i lleugera en ús de recursos que no requereix extensions HVM, però sí disposar de kernels i *drivers* amb suport de PV, de manera que l'hypervisor es pot executar de manera eficient sense emulació de màquina o emulació de components virtuals de maquinari.

- El suport de Paravirtualization. Els hostes d'aquest tipus són optimitzats per a poder executar-se com a màquina virtual (alguns estudis han proporcionat només sobrecàrregues entre el 2% i el 8%, mentre que en emulació les penalitzacions s'acostumen a apropar a un 20%), i donen millor rendiment que altres sistemes que es basen en components addicionals. A més, Xen fins i tot es pot executar en maquinari que no suporta extensions de virtualització.

Linux disposa del suport de paravirtualització des de la versió 2.6.23, anomenat *paravirt_ops* (o simplement *pvops*). Es tracta d'un kernel Linux de Domini 0 (disponible per a arquitectures x86, x86_64 i ia64). Aquest dom0 és el sistema que tenen els controladors de dispositiu per a comunicar-se amb el maquinari subjacent, executa les eines d'administració de Xen i proporciona els discos virtuals i el subsistema de xarxa virtual a la resta de sistemes hoste (denominats *domU*).

També cal esmentar que en les últimes generacions de processadors ha millorat sensiblement el suport i les possibilitats de les extensions HVM, la qual cosa ha fet viable noves aproximacions híbrides per a passar de virtualització PV a PVHVM; bàsicament consisteix en hoste HVM, però amb *drivers* especials per als discos i xarxa.

- Com que necessitem de base l'esmentat Domini 0, Xen necessitarà portar els sistemes operatius per a adaptar-se a l'API de Xen, a diferència de les VM tradicionals, que proporcionen entorns basats en programari per a simular el maquinari. En aquests moments hi ha ports (dels kernels pvops) per a NetBSD, FreeBSD i Linux, entre d'altres.
- Gràcies al codi aportat per Intel i AMD a Xen, s'han adaptat diferents extensions de virtualització de maquinari que permeten que els sistemes operatius sense modificacions s'executin en màquines virtuals Xen, la qual cosa ha aportat millores de rendiment i ha ofert la possibilitat d'executar sistemes GNU/Linux i Windows sense cap modificació.
- Hi ha suport per a la migració de màquines VM en calent entre equips físics. En aquest sentit, una màquina virtual en execució pot ser copiada i traslladada d'equip físic sense aturar l'execució, només per a petites sincronitzacions.

Enllaç d'interès

Una referència per a explicar el concepte de PVHVM:
http://www.slideshare.net/fullscreen/xen_com_mgr/linux-pv-on-hvm/

A continuació, veurem alguns usos simples de Xen sobre una distribució Debian, que disposa de la particularitat que és fàcil construir VM hostes basades en Debian de manera bastant flexible a partir dels repositoris de Debian.

Començarem per la instal·lació del domini 0 (`dom0`). Bàsicament, es fa amb una instal·lació Debian típica, en la qual únicament es tindran en compte les particions que cal emprar per a reservar espai posterior als hostes. En les versions actuals de Xen se sol reservar 4 GB de disc pel `dom0` (el seu `/`), i 1 GB de memòria per a la RAM que necessitarà. L'espai extra pot dedicar-se a emmagatzemar les VM hostes, o en el cas de *storage* podem, per exemple, fer servir volums LVM que ens permetran fer créixer els *filesystems* a mesura que els necessitem per part del `dom0` o les `domU` (hostes).

Per exemple, podríem fer 3 particions (`sda1,2,3`), i col·locar `/` (`root`) i `swap` en les dues primeres i un LVM en la tercera, amb un volum físic i un grup (`vg0`, per exemple). Instal·lem el sistema Debian.

Passem a instal·lar l'hypervisor mitjançant un metapaquet disponible en Debian que ens fa tot el treball de dependències i utilitats necessàries:

```
# apt-get install xen-linux-system
```

Podem comprovar també si la nostra màquina suporta extensions HVM, ja siguin d'Intel o AMD (en alguns casos poden estar desactivades en la BIOS de la màquina, i llavors haurem d'habilitar-les prèviament), amb:

```
egrep '(vmx|svm)' /proc/cpuinfo
```

Si estan disponibles, Xen podrà utilitzar més eficientment la paravirtualització, basant-se en el suport de virtualització dels processadors moderns. En diversos estudis (de *benchmarking*) s'ha comprovat que PV+HVM (sobre PV pura) en diferents *benchmarks* obté beneficis que van del 15% fins al 200% o 300%.

Un cop feta la instal·lació prèvia del sistema Xen, observarem que la distribució Debian, en la seva arrencada `Grub2` (en aquest cas), té una nova entrada denominada `Xen4`. És la que ens permetrà arrencar la màquina, sota l'hypervisor Xen gestionant el `dom0`. L'entrada Xen no està posada per defecte, amb la qual cosa se seguirà carregant per defecte el kernel del sistema. Amb aquestes instruccions, la posarem per defecte:

```
dpkg-divert --divert /etc/grub.d/08_linux_xen --rename /etc/grub.d/20_linux_xen
update-grub
```

Benchmarking PVHVM

Algunes referències:
<https://developer.rackspace.com/blog/welcome-to-performance-cloud-servers-have-some-benchmarks/> i
<https://xen-orchestra.com/debian-pvhvm-vs-pv/>

El següent pas serà configurar la xarxa per al domini 0. El més habitual és fer servir un *bridge* per programari (necessitem disposar del paquet *bridge-utils*); un exemple senzill per a `/etc/network/interfaces`:

```
#The loopback network interface
auto lo
iface lo inet loopback

iface eth0 inet manual

auto xenbr0
iface xenbr0 inet dhcp
    bridge_ports eth0

#other possibly useful options in a virtualized environment
#bridge_stp off          # disable Spanning Tree Protocol
#bridge_waitport 0      # no delay before a port becomes available
#bridge_fd 0            # no forwarding delay

## configure a (separate) bridge for the DomUs without
## giving Dom0 an IP on it
#auto xenbr1
#iface xenbr1 inet manual
#    bridge_ports eth1
```

Altres tècniques en la configuració són opcionals, com per exemple: ajustar l'ús de memòria per dom0, afegir a `/etc/default/grub` la següent línia per a reservar memòria per al dom0 (1 GB en aquest cas), i posteriorment un *update-grub*:

```
GRUB_CMDLINE_XEN="dom0_mem=1024M"
```

En Xen es fa servir una tècnica denominada *ballooned* que el que fa és assignar inicialment la majoria de memòria al dom0, i a mesura que van apareixent domU la va reduint. Amb això, la fixem de manera estàtica. Cal comprovar que sigui suficient; si patim algun *crash* del kernel Xen cal augmentar-la. També podem obviar aquesta personalització de memòria i deixar la tècnica de *balloning* per defecte. També cal fer el procés equivalent en el fitxer `/etc/xen/xend-config.sxp` amb:

```
(dom0-min-mem 1024)
(enable-dom0-ballooning no)
```

Amb aquests canvis, ja podrem arrencar la màquina de nou amb el dom0 de Xen disponible amb opcions bàsiques. La configuració de Xen en producció

escapa als termes d'espai d'aquesta secció, i recomanem consultar els manuals Xen per a configurar, especialment: a) les CPU disponibles a les màquines *guest* i *dom0*; b) comportament de *guests* en rearrencada; c) també és possible, per a depuració, activar la consola per port sèrie per a missatges de depuració.

Amb la configuració actual, passarem a detallar una instal·lació bàsica de domU (hoste) de tipus Debian, ja que se'ns ofereixen certes facilitats per al procés automàtic:

```
apt-get install xen-tools
```

Així, instal·lem *scripts* d'utilitats que ens proporciona Xen i després modificarem `/etc/xen-tools/xen-tools.conf` per a definir `dir=/mnt/disk` i `passwd=1`, amb l'adreça del pedaç on posarem les nostres màquines creades (en `/mnt/disk`, en el nostre cas un volum especial per a guardar les *guests*).

Amb l'ordre següent, i la configuració prèvia, podrem construir una imatge de Xen, una VM ja preparada amb la imatge de Debian corresponent (en aquest cas, una *wheezy*):

```
# xen-create-image --hostname nom_maquina --ip ip_maquina  
--vcpus 1 --pygrub --dist wheezy
```

Col·loquem en l'ordre anterior un nom de VM i una IP disponible (ja sigui pública o privada), i començarà la construcció de la imatge. En aquest cas, s'utilitzen els repositoris de Debian per a descarregar els paquets adequats per a construir la imatge.

Observarem que ens dóna les característiques per defecte de la màquina, i que ens comença a generar les imatges de les particions de *root* i *swap* en el directori (a partir del nostre cas `/mnt/disk`), `domains/nom_maquina` on residiran les imatges de la VM *guest* que s'està creant. Finalment ens demanarà el *password* de *root* de la màquina creada i finalitzarà:

```
General Information
```

```
-----
```

```
Hostname      : deb  
Distribution   : wheezy  
Mirror        : http://mirror.switch.ch/ftp/mirror/debian/  
Partitions    : swap          128Mb (swap)  
               /              4Gb   (ext3)  
Image type    : sparse  
Memory size   : 128Mb
```

```
Kernel path      : /boot/vmlinuz-3.2.0-4-amd64
Initrd path      : /boot/initrd.img-3.2.0-4-amd64
```

Networking Information

```
IP Address 1    : 10.0.0.2 [MAC: 00:16:3E:2D:D7:A2]
```

```
Creating partition image: /mnt/disk/domains/deb/swap.img
Creating swap on /mnt/disk/domains/deb/swap.img
Creating partition image: /mnt/disk/domains/deb/disk.img
Creating ext3 filesystem on /mnt/disk/domains/deb/disk.img
Installation method: debootstrap
Running hooks
Creating Xen configuration file
```

```
Setting up root password
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
All done
```

```
Logfile produced at:
/var/log/xen-tools/deb.log
```

Installation Summary

```
Hostname        : deb
Distribution     : wheezy
IP-Address(es)  : 10.0.0.2
RSA Fingerprint : 4e:f3:0a:47:c1:15:14:71:15:28:da:e2:9f:df:42:2b
Root Password   : N/A
```

Per a executar la màquina creada (*nombre_maquina=deb* en aquest exemple):

```
$ xm create /etc/xen/deb.cfg
```

I per a esborrar una imatge de VM:

```
$ xen-delete-image VMs_name
```

Per a fer una llista dels dominis actius:

```
$xm list
```

Name	ID	Mem	VCPUs	State	Time(s)
------	----	-----	-------	-------	---------

Domain-0	0	1023	1	r-----	247.9
deb	2	128	1	-b-----	5.4

I per a connectar una consola a la màquina disponible (al domini amb ID=2 en aquest cas):

```
$xm console 2
```

Fet que ens permet portar a terme la connexió a la màquina virtual i fer el procés de *login*.

7. Present del nucli i alternatives

Els avenços en el nucli de Linux en determinats moments van ser molt ràpids, però actualment, ja amb una situació bastant estable amb els nuclis de la branca 2.6.x i la branca posteriorment derivada 3.x, cada vegada passa més temps entre les versions que van apareixent. En certa manera això és força positiu: permet tenir temps per a corregir errors comesos, veure quines idees no han funcionat bé i provar-ne de noves, que, si tenen èxit, s'inclouen.

Comentarem en aquest apartat algunes de les idees dels últims nuclis i algunes que estan previstes, per a donar indicacions del que serà el futur pròxim en el desenvolupament del nucli.

En l'antiga branca 2.4.x del nucli [Ces06] es van fer algunes aportacions:

- Compliment dels estàndards IEEE POSIX, fet que permet que molts dels programes existents d'UNIX puguin recompilar-se i executar-se en Linux.
- Millor suport de dispositius: PnP, USB, port paral·lel, SCSI, etc.
- Suport per a nous sistemes de fitxers, com UDF (CD-ROM reescribibles com un disc). Altres sistemes amb *journal*, com els Reiser d'IBM o l'ext3, que permeten tenir un registre (*journal*) de les modificacions dels sistemes de fitxers i, així, poder recuperar-se d'errors o tractaments incorrectes dels fitxers.
- Suport de memòria fins a 4 GB. Al seu dia van sorgir alguns problemes (amb nuclis 1.2.x) que no suportaven més de 128 MB de memòria (una quantitat que en aquell temps era molta memòria).
- Millora de la interfície `/proc`. Es tracta d'un pseudosistema de fitxers (el directori `/proc`) que no existeix realment en disc, sinó que és simplement una manera organitzada d'accedir a dades del nucli i del maquinari.
- Suport del so en el nucli: es van afegir parcialment els controladors Alsa que abans es configuraven separatament.
- Es va incloure suport preliminar per al RAID programari i el gestor de volums dinàmics LVM1.

El nucli, en evolució

El nucli continua evolucionant, incorporant les últimes novetats en suport de maquinari i millores en les prestacions.

En l'antiga branca del nucli 2.6.x [Ces06, Pra03, Lov10], es va disposar d'importants avenços respecte a l'anterior (amb les diferents revisions .x de la branca 2.6):

- Millors prestacions en SMP, important per a sistemes multiprocessador molt utilitzats en entorns empresarials i científics.
- Millores en el planificador de CPU (*scheduler*). En particular, s'introdueixen avenços per a millorar l'ús de tasques interactives d'usuari, imprescindibles per a millorar l'ús de Linux en un ambient d'escriptori.
- Millores en el suport *multithread* per a les aplicacions d'usuari. S'incorporen nous models de fils NGPT (IBM) i NPTL (Red Hat) (amb el temps es va consolidar finalment l'NPTL).
- Suport per a USB 2.0 i, posteriorment, per a USB 3.0.
- Controladors Alsa de so incorporats en el nucli.
- Noves arquitectures de CPU de 64 bits: se suporten AMD x86_64 (també coneguda com a amd64) i PowerPC 64 i IA64 (arquitectura dels Intel Itanium).
- Sistemes de fitxers amb *journal*: JFS, JFS2 (IBM) i XFS (Silicon Graphics).
- Millores amb *journal* en els sistemes de fitxers propis, *ext3* i *ext4*, amb millores de la mida màxima dels arxius i de rendiment general.
- Millores de prestacions d'entrada/sortida i nous models de controladors unificats.
- Millores en la implementació de TCP/IP i el sistema NFSv4 (compartició de sistema de fitxers per xarxa amb altres sistemes).
- Millores significatives per a nucli preemptiu: permet que internament el nucli gestioni diverses tasques que es poden interrompre entre elles, característica imprescindible per a implementar eficaçment sistemes de temps real i també per a augmentar el rendiment de tasques interactives.
- Suspensió del sistema i restauració després de reiniciar (per nucli).
- UML, User Mode Linux, una mena de màquina virtual de Linux sobre Linux que permet veure un Linux (en mode usuari) executant-se sobre una màquina virtual. Això és ideal per a la depuració, ja que es pot desenvolupar i provar una versió de Linux sobre un altre sistema, i és útil tant per al mateix desenvolupament del nucli com per a una anàlisi de seguretat. En versions posteriors aquest concepte va evolucionar cap al mòdul KVM.

- Tècniques de virtualització incloses en el nucli: en les distribucions s'han anat incorporant diferents tècniques de virtualització, que necessiten extensions en el nucli. Podem destacar, per exemple, nuclis modificats per a Xen, Virtual Server (Vserver), OpenVZ o el mateix mòdul KVM.
- Nova versió del suport de volums LVM2.
- Nou pseudosistema de fitxers `/sys`, destinat a incloure la informació del sistema, i dispositius que aniran migrant des del sistema `/proc`, de manera que deixin aquest darrer amb informació relacionada amb els processos i el seu desenvolupament en execució, i també la informació dinàmica del mateix nucli.
- Mòdul FUSE per a implementar sistemes de fitxers en espai d'usuari (es fa servir en especial per al cas d'NTFS).

Per a conèixer els canvis de les versions més recents de Linux, poden examinar-se els fitxers `ChangeLog` que acompanyen cada versió del nucli en el seu codi font, o consultar un registre històric que es conserva a *Kernelnewbies.org*, en especial a l'adreça <http://kernelnewbies.org/LinuxChanges>, que manté els canvis de l'última versió, i poden consultar-se els de la resta de versions (en l'adreça <http://kernelnewbies.org/LinuxVersions>).

En les versions 3.x del kernel, bàsicament s'han millorat diferents aspectes de la branca prèvia 2.6.x, aportant noves prestacions, que també són el futur immediat de futures versions en el Kernel, les quals se centraran en:

- Increment de la tecnologia de virtualització en el nucli, per a suportar diferents configuracions de sistemes operatius i diferents tecnologies de virtualització, i un millor suport del maquinari per a virtualització inclòs en els processadors que sorgeixin en les noves arquitectures. Estan bastant suportades x86 i x86_64, amb KVM, per exemple, però n'hi ha d'altres que no ho estan o només ho estan parcialment.
- El suport d'SMP (màquines multiprocessador), de CPU de 64 bits (Xeon, nous *multicore* d'Intel i Opteron d'AMD), el suport de CPU *multicore* i l'escalabilitat d'aplicacions multifil en aquestes CPU.
- La millora de sistemes de fitxers per clusterització i grans sistemes distribuïts.
- Millores en sistemes de fitxers estàndard Linux per a adaptar-los a noves necessitats, com el cas de Btrfs i diverses millores introduïdes en ext4, i també les millores introduïdes i millor suport en el kernel per XFS i ZFS.
- Per contra, la millora en nuclis més optimitzats per a dispositius mòbils (*smartphones*, *tablets*, etc.). Per exemple, s'ha incorporat el nucli d'Android,

com a plataforma, al codi font del kernel. I hi ha diverses iniciatives com Ubuntu Phone o Tizen, per a proporcionar noves plataformes basant-se en kernel Linux per a entorns mòbils. Especialment, el suport del kernel per a arquitectures ARM (de les més utilitzades en entorns mòbils) ha portat la possibilitat d'usar Linux en multitud de nous dispositius.

- Millora en el compliment dels estàndards POSIX.
- Millora de la planificació de la CPU. Encara que es van fer molts avenços en aquest aspecte, encara hi ha un baix rendiment en algunes situacions. En particular, en l'ús d'aplicacions interactives d'escriptori s'estan estudiant diferents alternatives per a millorar aquest i altres aspectes relacionats amb l'escriptori i l'ús del rendiment gràfic.
- Suport per a les noves EFI/UEFI com a substitutes de les antigues BIOS en entorns de PC x86/x86_64.
- Suport per a un nou sistema de filtratge IP NFtables, que substituirà progressivament els tallafocs mitjançant iptables.

També, malgrat que s'aparta dels sistemes Linux, la Free Software Foundation (FSF) i el seu projecte GNU continuen treballant en el projecte d'acabar un sistema operatiu complet. Cal recordar que el projecte GNU tenia com a principal objectiu aconseguir un clon UNIX de programari lliure, i les utilitats GNU només són el programari de sistema necessari. A partir de 1991, quan Linus aconsegueix conjuntar-ne el nucli amb algunes utilitats GNU, es va fer un primer pas que ha acabat en els sistemes GNU/Linux actuals. Però el projecte GNU continua treballant en la idea d'acabar el sistema complet. En aquest moment disposen ja d'un nucli en el qual poden córrer les utilitats GNU. Aquest nucli s'anomena Hurd, i un sistema construït amb aquest nucli es coneix com a GNU/Hurd. Ja existeixen algunes distribucions de prova, en concret, una Debian GNU/Hurd.

Hurd va ser pensat com el nucli per al sistema GNU cap al 1990, quan en va començar el desenvolupament, ja que llavors la major part del programari GNU ja estava desenvolupat i només en faltava el nucli. Va ser el 1991 quan Linus va combinar GNU amb el nucli Linux i va crear així l'inici dels sistemes GNU/Linux. Però Hurd continua en procés de desenvolupament. Les idees de desenvolupament a Hurd són més complexes, ja que Linux podria considerar-se un disseny “conservador” que parteix d'idees ja conegudes i implantades.

En concret, Hurd estava pensat com una col·lecció de servidors implementats sobre un micronucli Mach [Vah96], que és un disseny de nucli tipus micronucli (a diferència de Linux, que és de tipus monolític) desenvolupat per la Universitat Carnegie Mellon i posteriorment per la Universitat d'Utah. La idea bàsica era modelitzar les funcionalitats del nucli d'UNIX com a servidors que s'implementarien sobre un nucli bàsic Mach. El desenvolupament de Hurd es

Enllaç d'interès

Per a saber més sobre POSIX, podeu visitar el següent web:
<http://www.unix.org/>

Enllaç d'interès

Per a saber més sobre el projecte GNU podeu visitar el web:
<http://www.gnu.org/gnu/thegnuproject.html>

Enllaç d'interès

Podeu llegir les opinions de Richard Stallman sobre GNU i Linux a:
<http://www.gnu.org/gnu/linux-and-gnu.html>

va retardar mentre s'estava acabant el disseny de Mach, i aquest es va publicar finalment com a programari lliure, cosa que permetria fer-lo servir per a desenvolupar Hurd. En aquest punt hem de comentar la importància de Mach, ja que molts sistemes operatius s'han basat en idees extremes d'aquest nucli, el més destacat dels quals és el MacOS X d'Apple.

El desenvolupament de Hurd es va retardar més per la complexitat interna, ja que existien diversos servidors amb diferents tasques de tipus *multithread* (d'execució de múltiples fils) i la depuració era extremadament difícil. Però avui en dia es disposa d'algunes versions de prova, i també de versions de prova de distribució GNU/Hurd produïdes per Debian. Tot i així, el projecte en si mateix no és especialment optimista respecte a obtenir sistemes en producció, a causa tant de la complexitat com de la manca de suport per a dispositius.

Potser en un futur no tan llunyà hi podrà haver avenços i coexistència de sistemes GNU/Linux amb GNU/Hurd, o fins i tot el nucli Linux serà substituït pel Hurd si es fan avenços importants en el seu desenvolupament. Això seria una solució si en algun moment Linux s'estanca (ja que el seu disseny monolític pot causar problemes si es fa molt més gran). En qualsevol cas, tant uns sistemes com els altres tenen un futur prometedor davant seu. El temps dirà cap a on s'inclina la balança.

8. Taller de configuració del nucli a les necessitats de l'usuari

En aquest apartat, veurem un petit taller interactiu per al procés d'actualització i configuració del nucli en el parell de distribucions utilitzades: Debian i Fedora.

Una primera cosa imprescindible, abans de començar, és conèixer la versió actual que tenim del nucli, mitjançant `uname -r`, per a poder determinar quina és la versió següent que volem actualitzar o personalitzar. I una altra és la de disposar de mitjans per a arrencar el nostre sistema en cas de fallades: el conjunt de CD/DVD de la instal·lació, el disquet (o CD) de rescat (actualment s'acostuma a fer servir el primer CD/DVD de la distribució) o alguna distribució en LiveCD que ens permeti accedir al sistema de fitxers de la màquina, per a refer configuracions que hagin causat problemes. A més, hauríem de fer una còpia de seguretat de les nostres dades o configuracions importants.

Veurem les possibilitats següents:

- 1) Actualització del nucli de la distribució. Cas automàtic de Debian.
- 2) Actualització automàtica en Fedora.
- 3) Personalització d'un nucli genèric (tant Debian com Fedora). En aquest últim cas, els passos són bàsicament els mateixos que els que es presenten en l'apartat de configuració, però farem alguns comentaris addicionals.

8.1. Configuració del nucli en Debian

En el cas de la distribució Debian, la instal·lació es pot fer també de manera automàtica mitjançant el sistema de paquets d'APT. Pot fer-se tant des de la línia d'ordres com amb gestors APT gràfics (synaptic, per exemple).

Farem la instal·lació per línia d'ordres amb `apt-get`, suposant que l'accés als paquets font apt (sobretot en els Debian originals) està ben configurat en el fitxer de `/etc/apt/sources.list`. Vegem els passos:

- 1) Actualitzar la llista de paquets:

```
# apt-get update
```

- 2) Fer una llista dels paquets associats a imatges del nucli:

```
# apt-cache search linux-image
```

3) Triar una versió adequada a la nostra arquitectura (genèrica, x86 o i386 per a Intel o AMD o, en particular per a 64 bits, versions amd64 per a Intel i AMD). El codi de la versió indica la versió del nucli, la revisió de Debian del nucli i l'arquitectura. Per exemple, 3.14-1-amd64 és un nucli per a x86_64 AMD/Intel, revisió Debian 1 del nucli 3.14.

4) Comprovar, per a la versió escollida, que hi ha els mòduls accessoris extres. Amb `apt-cache` busquem si hi ha altres mòduls dinàmics que puguin ser interessants per al nostre maquinari, segons la versió del nucli que cal instal·lar. Recordeu que, com vam veure en la *Debian Way*, també trobem la utilitat `module-assistant`, que ens permet automatitzar aquest procés si els mòduls estan suportats. En el cas en què els mòduls necessaris no estiguessin suportats, això ens podria impedir actualitzar el nucli si considerem que el funcionament del maquinari problemàtic és vital per al sistema.

5) Buscar, si volem disposar també del codi font del nucli, els `linux-source-version` (en aquest cas ha de ser 3.14, és a dir, el número principal) i els `linux-headers` corresponents, per si més tard volem fer un nucli personalitzat (en aquest cas, el nucli genèric corresponent pedaç per a Debian).

6) Instal·lar el que hàgim decidit. Si volem compilar des dels paquets font o simplement disposar del codi:

```
# apt-get install linux-image-version
```

(si fossin necessaris alguns mòduls) i

```
# apt-get install linux-source-version-generica
# apt-get install linux-headers-version
```

7) Instal·lar el nou nucli, per exemple en el *bootloader* LILO o Grub (en les últimes versions trobarem aquest per defecte). Normalment això es fa automàticament, però no estaria de més fer alguna còpia de seguretat prèvia de la configuració del *bootloader* (ja sigui `/etc/lilo.conf` o `/boot/grub/menu.lst` o `grub.cfg`).

Si se'ns pregunta si tenim el `initrd` activat, caldrà verificar el fitxer de LILO (`/etc/lilo.conf`) i incloure la nova línia en la configuració LILO de la imatge nova:

```
initrd = /initrd.img-version (o /boot/initrd.img-version)
```

Una vegada feta aquesta configuració, hauríem de tenir un LILO semblant al següent llistat (fragment del fitxer), suposant que `initrd.img` i `vmlinuz` siguin enllaços a la posició dels fitxers del nou nucli:

```
default = Linux
image = /vmlinuz
label = Linux
initrd = /initrd.img
```

```
# restricted
# alias = 1
image = /vmlinuz.old
    label = LinuxOLD
    initrd = /initrd.img.old
# restricted
# alias = 2
```

Tenim la primera imatge per defecte, i l'altra és el nucli antic. Així doncs, des del menú LILO podrem demanar-ne una o una altra o, simplement canviant el default, recuperar l'antiga. Sempre que fem modificacions en l'arxiu `/etc/lilo.conf` no hem d'oblidar-nos de reescriure'l en el sector corresponent amb l'ordre `/sbin/lilo o /sbin/lilo -v`.

En el cas de Grub, que acostuma a ser l'opció normal en les distribucions (ara Grub2 en particular), s'hauria creat una nova entrada (cal tenir present fer la còpia de seguretat prèvia, perquè podem haver perdut alguna entrada anterior depenent del funcionament o configuració de Grub; per exemple, pot limitar-se el nombre màxim d'entrades):

```
title                Debian GNU/Linux, kernel 2.6.32-5-amd64
root                 (hd0,0)
kernel               /boot/vmlinuz-2.6.32-5-amd64 \
                    root=UUID=4df6e0cd-1156-444e-bdfd-9a9392fc3f7e ro
initrd               /boot/initrd.img-2.6.32-5-amd64
```

on apareixerien els fitxers binaris del nucli i `initrd`. Amb l'etiqueta `root` en el nucli apareix l'identificador de la partició arrel del sistema on està instal·lat el nucli, identificador que és comú a totes les entrades de nuclis del mateix sistema. Abans s'utilitzava un esquema amb `root=/dev/hda0 o /dev/sda0`, però aquest esquema ja no és útil, perquè en la detecció dels discos pot canviar l'ordre d'aquests en el sistema; així, es prefereix etiquetar les particions. Aquestes etiquetes es poden obtenir mitjançant l'ordre del subsistema `udisks` `udisksctl info -b /dev/particion, vegeu symlinks`, o també amb l'ordre `dumpe2fs /dev/particion | grep UUID` o, si la partició es troba muntada en el sistema, consultant `/etc/fstab`.

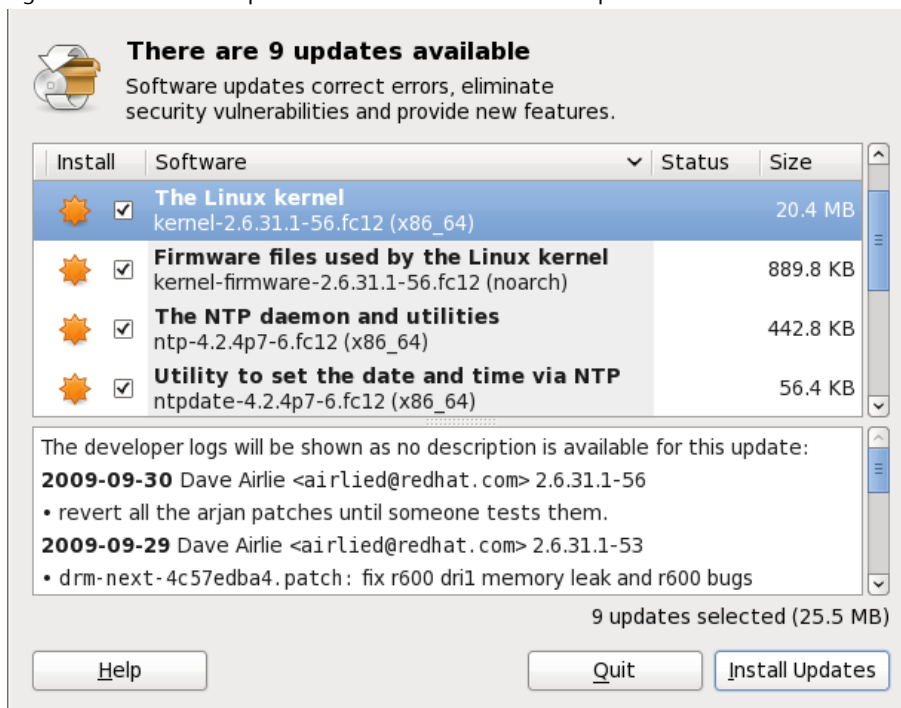
8.2. Configuració del nucli en Fedora/Red Hat

L'actualització del nucli en la distribució Fedora/Red Hat és totalment automàtica per mitjà del seu servei de gestió de paquets (`yum` en línia d'ordres, per exemple), o bé mitjançant els programes gràfics que inclou la distribució, depenent de la seva versió, per a l'actualització (`PackageKit` en Fedora o `pup` en Red Hat empresarial o equivalents com CentOS). Normalment, les trobarem en la barra de tasques o en el menú d'eines de sistema de Fedora/Red Hat.

Aquest programa d'actualització bàsicament verifica els paquets de la distribució actual enfront d'una base de dades de Fedora/Red Hat, i ofereix la possibilitat de descarregar els paquets actualitzats, entre aquests els del nucli. Aquest servei, en el cas de Red Hat empresarial, funciona per un compte de servei i Red Hat l'ofereix per pagament. Amb aquest tipus d'utilitats, l'actualització del nucli és automàtica. Cal comprovar les utilitats disponibles en els menús Eines/Administració, ja que les eines gràfiques disponibles en una distribució són altament dependents de la seva versió, perquè són actualitzades amb freqüència.

Per exemple, en la figura 8, observem que una vegada posat en execució ens ha detectat una nova versió del nucli disponible i podem seleccionar-la perquè ens la descarregui.

Figura 8. Eina de Fedora que mostra l'actualització del nucli disponible



En Fedora podem utilitzar les eines gràfiques equivalents o fer servir directament yum, si coneixem la disponibilitat de nous nuclis:

```
# yum install kernel kernel-source
```

Una vegada descarregada, es procedirà a la seva instal·lació, normalment també de manera automàtica, ja que disposem de Grub o LILO com a gestors d'arrencada. En el cas de Grub, sol ser automàtic i deixa un parell d'entrades en el menú, una per a la versió més nova i una altra per a l'antiga. Per exemple, en aquesta configuració de Grub (el fitxer està en /boot/grub/grub.cfg o bé /boot/grub/menu.lst), tenim dos nuclis diferents, amb els números respectius de versió:

```
#fichero grub.conf
default = 1
timeout = 10
splashimage = (hd0,1)/boot/grub/splash.xpm.gz

title Linux (2.6.30-2945)
root (hd0,1)
kernel /boot/vmlinuz-2.6.30-2945 ro
        root = UUID=4df6e0cd-1156-444e-bdfd-9a9392fc345f
initrd /boot/initrd-2.6.30-18.9.img

title LinuxOLD (2.6.30-2933)
root (hd0,1)
kernel /boot/vmlinuz-2.4.30-2933 ro
        root = UUID=4df6e0cd-1156-444e-bdfd-9a9392fc345f
initrd /boot/initrd-2.4.30-2933.img
```

Observeu que la configuració actual és Grub-Legacy; per a Grub2, els camps són semblats però fa falta consultar les *menuentry* corresponents a cada opció.

Cada configuració inclou un títol, que apareixerà en l'arrencada; el *root*, o partició del disc des d'on arrencar; el directori on es troba el fitxer corresponent al nucli i el fitxer *initrd* corresponent.

En cas que disposem de LILO* com a gestor en la Fedora/Red Hat, el sistema també l'actualitza (fitxer */etc/lilo.conf*), però després caldrà reescriure l'arrencada amb l'ordre */sbin/lilo -v* manualment.

*Per defecte, es fa servir Grub.

Cal assenyalar, així mateix, que amb la instal·lació anterior teníem possibilitats de descarregar els paquets font del nucli; aquests, una vegada instal·lats, són a */usr/src/linux-version*, i es poden configurar i compilar pel procediment habitual, com si fossin un nucli genèric. Cal esmentar que l'empresa Red Hat duu a terme un gran treball de pedaços i correccions per al nucli (usat després en Fedora), i que els seus nuclis són modificacions de l'estàndard genèric amb bastants afegits, per la qual cosa pot ser millor utilitzar les fonts pròpies de Red Hat, tret que vulguem un nucli més nou o experimental que aquell que ens proporcionen.

8.3. Configuració d'un nucli genèric

Vegem el cas general d'instal·lació d'un nucli a partir de les seves fonts. Suposem que tenim unes fonts ja instal·lades en */usr/src* (o el prefix corresponent).

Normalment, tindrem un directori *linux*, *linux-version* o senzillament el nombre versió; aquest serà l'arbre dels paquets font del nucli. Aquests poden provenir de la mateixa distribució (o pot ser que els hàgim descarregat d'una

actualització prèvia), i en primer lloc serà interessant comprovar si són els últims disponibles, com ja hem fet abans amb Fedora o Debian. D'altra banda, si volem tenir les últimes i genèriques versions, podem anar a *kernel.org* i descarregar l'última versió disponible (millor l'estable que les experimentals, tret que estiguem interessats en el desenvolupament del nucli). Descarreguem l'arxiu i descomprimim en `/usr/src` (o un altre d'escollit, potser millor) els paquets font del nucli. També podríem buscar si hi ha pedaços per al nucli i aplicar-los (segons hem comentat en l'apartat 4).

A continuació comentarem els passos que caldrà fer. El procediment que s'indica en aquesta part del taller és genèric, però pot donar algun problema depenent de la distribució usada. Es recomana seguir en tant que sigui possible el subapartat 3.1., on es comenta el cas de configuració d'un nucli *vanilla*, o bé els comentaris en el cas Debian, en el subapartat 3.2., o la referència [Fedk] per al cas Fedora.

Amb les consideracions esmentades, podem seguir també el procés següent:

1) Netejar el directori de proves anteriors (si és el cas):

```
make mrproper
```

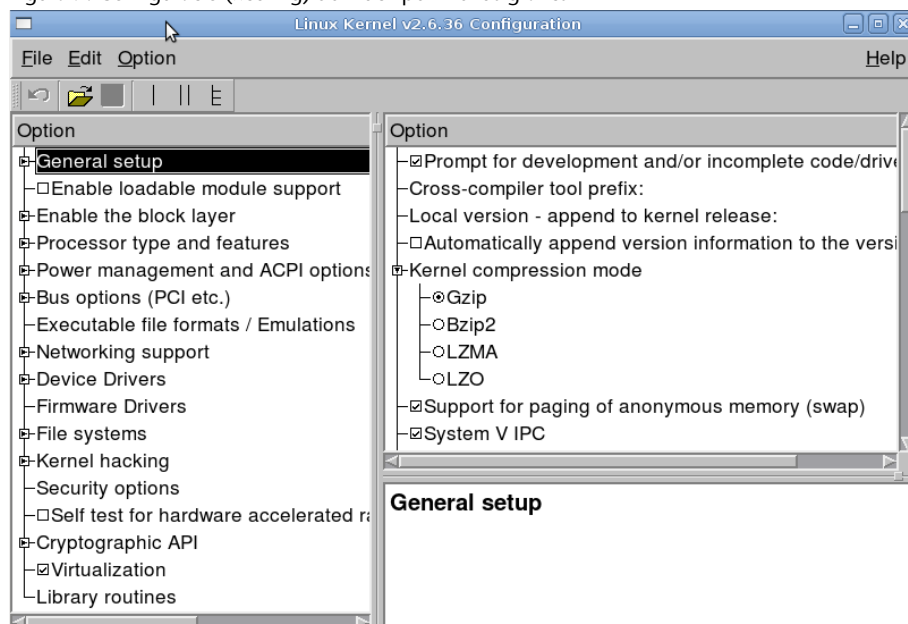
2) Configurar el nucli, per exemple, amb `make menuconfig` (o `xconfig`, figura 9, `gconfig` o `oldconfig`). Ho vam veure en el subapartat 3.1.

3) Compilar i crear la imatge del nucli: `make`. El procés pot durar desenes de minuts en maquinari modern o diverses hores en maquinari molt antic. Quan finalitza, la imatge és a `/usr/src/directori-fonts/arch/i386/boot` (en el cas d'arquitectura Intel de 32/64 bits).

Vegeu també

Seria convenient rellegir l'apartat 3.

Figura 9. Configuració (xconfig) del nucli per menús gràfics



4) Ara compilem els mòduls amb `make modules`. Fins a aquest moment, no hem modificat res en el nostre sistema. Ara haurem de procedir a la instal·lació.

Tanmateix, també s'ha d'anar amb compte si estem compilant una versió que és la mateixa (exacta numeració) que la que tenim (els mòduls se sobreescriran); en aquest cas, és millor fer una còpia de seguretat dels mòduls:

```
cd /lib/modules
tar -cvzf old_modules.tgz versionkernel-antigua/
```

Així, tenim una versió en `.tgz` que podríem recuperar després en cas de problemes. Finalment, instal·lem els mòduls amb:

```
# make modules install
```

5) Ara podem passar a la instal·lació del nucli, per exemple (de manera manual) amb:

```
# cd /usr/src/directorio-Fuentes/arch/i386/boot
# cp bzImage /boot/vmlinuz-versionkernel
# cp System.map /boot/System.map-versionkernel
# ln -s /boot/vmlinuz-versionkernel /boot/vmlinuz
# ln -s /boot/System.map-versionkernel /boot/System.map
```

Així col·loquem el fitxer de símbols del nucli (`System.map`) i la imatge del nucli. Cal recordar que pot ser necessària també una imatge de `initrd`.

6) Ja només ens queda posar la configuració necessària en el fitxer de configuració del gestor d'arrencada, ja sigui LILO (`/etc/lilo.conf`), Grub-Legacy o Grub2 (`/boot/grub/menu.lst` o `grub.cfg`) segons les configuracions que ja vam veure amb Fedora o Debian. I recordeu que en el cas de LILO caldrà tornar a actualitzar la configuració amb `/sbin/lilo` o `/sbin/lilo -v`.

Cal recordar que, com vam veure, tot aquest procés en els kernels moderns pot fer-se simplement amb un:

```
# make install
```

7) Reiniciar la màquina i observar els resultats (si tot el procés ha estat correcte).

Resum

En aquest mòdul s'han examinat diferents característiques del nucli (*kernel*) de Linux en les seves branques 2.6 i 3.x. Així mateix, s'han comentat alguns dels processos d'actualització, configuració i sintonització per al sistema, útils tant per a l'optimització de l'ús de memòria com per a maximitzar les prestacions en una arquitectura de CPU concreta, i l'adaptació al maquinari (dispositius) present en el sistema.

També hem examinat les possibilitats que ofereixen els mòduls dinàmics com a mecanisme d'extensió del nucli i ampliació del maquinari suportat.

La inclusió en el nucli de tècniques de virtualització ens permet, a partir del nucli i certes utilitats, construir un entorn de virtualització potent i flexible per a generar diferents combinacions de màquines virtuals que resideixen en un sistema amfitrió GNU/Linux.

Activitats

1. Determineu la versió actual del nucli Linux incorporada en la vostra distribució. Comproveu les actualitzacions disponibles de manera automàtica, tant a Debian (apt) com a Fedora/Red Hat (mitjançant yum).
2. Feu una actualització automàtica de la vostra distribució. Comproveu possibles dependències amb altres mòduls utilitzats i amb el *bootloader* (LILO o Grub-Legacy o Grub2) utilitzat. En funció del sistema, pot ser recomanable una còpia de seguretat de les dades importants del sistema (comptes d'usuaris i fitxers de configuració modificats), o bé fer el procés en un altre sistema de què es disposi per a proves.
3. Per a la vostra branca del nucli, determineu l'última versió disponible (consulteu la pàgina web <http://www.kernel.org>) i feu una instal·lació manual amb els passos examinats en el mòdul. La instal·lació final pot deixar-se com a opcional, o bé posar una entrada dins del *bootloader* per a les proves del nucli nou. Considereu, també en funció del sistema, la possibilitat de fer una còpia de seguretat prèvia, en especial de les configuracions estables dels *bootloaders*.
4. En el cas de la distribució Debian, a més dels passos manuals, hi ha, com hem vist, una manera especial (recomanada) d'instal·lar el nucli a partir de les seves fonts mitjançant el paquet `kernel-package`. Feu els passos necessaris per a crear una versió personalitzada d'un nucli *vanilla*.
5. Elaboreu una màquina virtual basada en KVM que tingui com a hoste una altra distribució GNU/Linux diferent de la del sistema amfitrió, a partir de la seva instal·lació per mitjà de CD-ROM o per mitjà d'una imatge ISO de la distribució.

Bibliografia

- [Arc] **Arcomano, R.** *Kernel Analysis-HOWTO*. The Linux Documentation Project.
- [Bac86] **Bach, M. J.** (1986). *The Design of the UNIX Operating System*. Prentice Hall.
- [Ces06] **Cesati, M.; Bovet, D.** (2006). *Understanding the Linux Kernel* (3a. ed.). O'Reilly.
- [Cor05] **Corbet, J.; Rubini, A.; Kroah-Hartman, G.** (2005). *Linux Device Drivers* (3a. ed.). O'Reilly.
- [Debk] **Debian Kernel Handbook Project.** *Debian Linux Kernel Handbook*.
<<http://kernel-handbook.alioth.debian.org>>
- [Fedk] **Fedora Project.** *Building a custom kernel*.
<http://fedoraproject.org/wiki/Building_a_custom_kernel>
- [Gor] **Gortmaker, P.** (2003). *The Linux BootPrompt HOWTO*. The Linux Documentation Project.
- [Grub1] **GNU.** *Grub bootloader*.
<<http://www.gnu.org/software/grub/grub-legacy.html>>
- [Grub2] **GNU.** *Grub Manual*.
<<http://www.gnu.org/software/grub/manual/>>
- [Hen] **Henderson, B.** *Linux Loadable Kernel Module HOWTO*. The Linux Documentation Project.
- [Kan] **Kanis, I.** *Multiboot with GRUB Mini-HOWTO*. The Linux Documentation Project.
- [Ker] **Rusty Russell.** *Unreliable Guide To Hacking The Linux Kernel*.
<<http://www.kernel.org/doc/html/docs/kernel-hacking/index.html>>
- [Kera] **Kernelnewbies.org.** *Kernel Newbies*.
<<http://www.kernelnewbies.org>>
- [Kerb] **Kernel.org.** *Linux Kernel Archives*.
<<http://www.kernel.org>>
- [Lkm] **Lkm.** *Linux Kernel Mailing List*.
<<http://www.tux.org/lkml>>

- [Lov10] **Love, R.** (2010). *Linux Kernel Development*. (3a. ed.). Addison-Wesley.
- [Mur] **Murphy, G. L.** *Kernel Book Project*.
<<http://kernelbook.sourceforge.net>>
- [OSDa] **OSDL**. *Open Source Development Laboratories*. (Ara *The Linux Foundation*)
<<http://www.linuxfoundation.org>>
- [Pra03] **Pranevich, J.** (2003). *The Wonderful World of Linux 2.6*.
<<http://www.kniggit.net/wwol26.html>>
- [Pra11] **Pranevich, J.** (2011). *The Wonderful World of Linux 3.0*.
<<http://www.kniggit.net/wwol30/>>
- [Skoa] **Skoric, M.** *LILO mini-HOWTO*. The Linux Documentation Project.
- [Tan87] **Tanenbaum, A.** (1987). *Sistemas operativos: Diseño e Implementación*. Prentice Hall.
- [Tan06] **Tanenbaum, A.; Woodhull, A. S.** (2006). *Operating Systems Design and Implementation* (3a. ed.). Prentice Hall.
- [Tum] **Tumenbayer, E.** (2002). *Linux SMP HOWTO*. The Linux Documentation Project.
- [Vah96] **Vahalia, U.** (1996). *UNIX Internals: The New Frontiers*. Prentice Hall.
- [Vasb] **Vasudevan, A.** *The Linux Kernel HOWTO*. The Linux Documentation Project.
- [Zan] **Zanelli, R.** *Win95 + WinNT + Linux multiboot using LILOmini-HOWTO*. The Linux Documentation Project.

Sobre aquestes fonts de referència i informació:

[Kerb] Lloc que proporciona un repositori de les diverses versions del nucli Linux i els seus pedaços.

[Kera] [lkm] Llocs web que inclouen una part de la comunitat del nucli de Linux. Disposa de diversos recursos de documentació i llistes de correu de l'evolució del nucli, la seva estabilitat i les noves prestacions que es desenvolupen.

[Debk] És un manual imprescindible sobre els processos de compilació del nucli en la distribució Debian. S'acostuma a actualitzar amb els canvis produïts en la distribució. [Fedk] aporta una referència similar per al cas Fedora.

[Ces06] Llibre sobre el nucli de Linux 2.4, que detalla els diferents components i la seva implementació i disseny. Hi ha una primera edició sobre el nucli 2.2 i una nova actualització al nucli 2.6. [Lov10] és un text alternatiu més actualitzat. Per a la branca 3.x aquests llibres continuen essent útils, ja que la majoria de conceptes del nucli es mantenen inalterats.

[Pra03] Article que descriu algunes de les principals novetats de la branca 2.6 del nucli Linux. [Pra11] és l'equivalent per a 3.0+.

[Ker] [Mur] Projectes de documentació del nucli, incomplets però amb material útil.

[Bac86] [Vah96] [Tan87] [Tan86] Alguns textos sobre els conceptes, disseny i implementació dels nuclis de diferents versions UNIX i Linux.

[Skoa][Zan01][Kan][Grub1][Grub2] Recursos per a tenir més informació sobre els carregadors LILO, Grub i Grub2.

[Gru2][Grub1] Llocs oficials de Grub2 i l'anterior original Grub (ara conegut com a Grub Legacy.)

Administració de servidors

Remo Suppi Boldrito

PID_00212465

Índex

Introducció	5
Objectius	6
1. Administració de servidors	7
1.1. Sistema de noms de domini (<i>Domain Name System, DNS</i>)	7
1.1.1. Servidor de noms cau	8
1.1.2. Configuració d'un domini propi	10
1.1.3. Altres DNS	14
1.2. NIS (YP)	16
1.2.1. Com es pot iniciar un client local de NIS en Debian?	17
1.2.2. Quins recursos s'han d'especificar per a utilitzar el NIS?	18
1.2.3. Com s'ha de configurar un servidor?	19
1.3. Serveis de connexió remota: Telnet i ssh	21
1.3.1. Telnet i telnetd	21
1.3.2. SSH, <i>Secure shell</i>	22
1.3.3. VPN SSL (via <i>tun driver</i>)	25
1.3.4. Túnel encadenats	26
1.4. Serveis de transferència de fitxers: FTP	27
1.4.1. Client FTP (convencional)	27
1.4.2. Servidors FTP	28
1.5. <i>Active Directory Domain Controller</i> amb Samba4	30
1.5.1. Passos preliminars	31
1.5.2. Compilar Samba4	32
1.6. Serveis d'intercanvi d'informació en nivell d'usuari	36
1.6.1. El <i>Mail Transport Agent</i> (MTA)	36
1.6.2. External SMTP	37
1.7. <i>Internet Message Access Protocol</i> (IMAP)	38
1.7.1. Aspectes complementaris	39
1.8. Grups de discussió	41
1.9. <i>World Wide Web</i> (httpd)	42
1.9.1. Servidors virtuals	44
1.9.2. Apache + PHP + Mysql + PhpMyAdmin	46
1.9.3. Altres servidors httpd	47
1.10. Servidor de WebDAV	49
1.11. Servei de <i>proxy</i> : Squid	51
1.11.1. <i>Proxy SOCKS</i>	54
1.12. OpenLdap (LDAP)	56
1.13. Serveis d'arxius (NFS, <i>Network File System</i>)	60
1.14. Servidor de wiki	61
1.14.1. Instal·lació ràpida	62

1.14.2. Instal·lació de servidor.....	62
1.15. Gestió de còpies de seguretat (<i>backups</i>).....	64
1.15.1. Programes habituals de còpies de seguretat.....	64
1.15.2. rdiff-backup i rdiff-backups-fs	66
1.16. <i>Public Key Infrastructure</i> (PKI)	67
Activitats	72
Bibliografia	72

Introducció

La interconnexió entre màquines i les comunicacions d'alta velocitat han permès que els recursos que s'utilitzin no estiguin al mateix lloc geogràfic de l'usuari. UNIX (i per descomptat, GNU/Linux) és probablement el màxim exponent d'aquesta filosofia, ja que des del seu inici ha fomentat l'intercanvi de recursos i la independència de dispositius. Aquesta filosofia s'ha plasmat en una cosa comuna avui dia com són els serveis. Un servei és un recurs (que pot ser universal o no) que permet, sota certes condicions, obtenir informació, compartir dades o simplement processar la informació a distància. El nostre objectiu és analitzar els serveis que permeten el funcionament d'una xarxa. Generalment, dins d'aquesta xarxa hi haurà una màquina (o més, segons les configuracions) que farà possible l'intercanvi d'informació entre les altres. Aquestes màquines es denominen *servidors* i contenen un conjunt de programes que permeten que la informació estigui centralitzada i sigui fàcilment accessible. Aquests serveis permeten la reducció de costos i amplien la disponibilitat de la informació, però s'ha de tenir en compte que un servei centralitzat presenta inconvenients, ja que pot quedar fora de servei i deixar sense atenció tots els usuaris. En aquest mòdul, es veuran els principals serveis que permeten que una màquina GNU/Linux jugui un paper molt important en una infraestructura tecnològica, tant a centralitzar i distribuir dades com a ser punt d'informació, accés o comunicació. D'altra banda, i amb l'avenç de les arquitectures (programari) orientades a serveis (SOA - *Service Oriented Architecture*), i les tecnologies de desenvolupament d'aplicacions que s'han estandarditzat en aquest paradigma de disseny de sistemes distribuïts, GNU/Linux s'ha transformat en la infraestructura per excel·lència que dona suport a la creació de sistemes d'informació altament escalables. Aquest tipus d'arquitectura (SOA) s'ha transformat en una part essencial del desenvolupament de programari distribuït, ja que permet la creació de sistemes distribuïts eficients, que aprofiten tota la infraestructura subjacent, i estableix una interfície ben definida a l'exposició i crida de serveis web (de manera comuna però no exclusivament), cosa que facilita la interacció entre els sistemes propis i externs.

Serveis replicats

Una arquitectura de servidors ha de tenir els serveis replicats (*mirrors*) per a solucionar els inconvenients que comporta.

Objectius

En els materials didàctics d'aquest mòdul, trobareu els continguts i les eines procedimentals per aconseguir els objectius següents:

- 1.** Presentar els aspectes més rellevants dels conceptes involucrats, tant en un nivell teòric com pràctic, en l'estructura de servidors/serveis en un sistema GNU/Linux.
- 2.** Analitzar els conceptes relatius a serveis i servidors específics d'un sistema GNU/Linux.
- 3.** Experimentar amb la configuració i adaptar la instal·lació de serveis a un entorn determinat.
- 4.** Analitzar i participar en discussions sobre les possibilitats actuals i futures de nous serveis i els obstacles que hi ha bàsicament en aspectes de seguretat en els diferents entorns de treball GNU/Linux (servidor, escriptori multimèdia, escriptori ofimàtica, encaminador o *router*, etc.).

1. Administració de servidors

Els serveis es poden classificar en dos tipus: de vinculació ordinador-ordinador o de relació home-ordinador. En el primer cas, es tracta de serveis requerits per altres ordinadors, mentre que, en el segon, són serveis requerits pels usuaris (encara que hi ha serveis que poden actuar en ambdues categories). Dins del primer tipus es troben serveis de noms, com el *Domain Name System* (DNS), el servei d'informació d'usuaris (NIS-YP), el directori d'informació LDAP o els serveis d'emmagatzematge intermedi (*proxies*). Dins de la segona categoria es preveuen serveis de connexió interactiva i execució remota (ssh, Telnet), transferència de fitxers (ftp), intercanvi d'informació en nivell d'usuari, com el correu electrònic (MTA, IMAP, POP), *news*, *World Wide Web*, *wiki* i arxius (NFS). Per a mostrar les possibilitats de GNU/Linux Debian-FC, es descriurà cadascun d'aquests serveis amb una configuració mínima i operativa, però sense descuidar aspectes de seguretat i estabilitat.

1.1. Sistema de noms de domini (*Domain Name System, DNS*)

La funcionalitat del servei de DNS és convertir noms de màquines (llegibles i fàcils de recordar pels usuaris) en adreces IP o viceversa.

A la consulta de quina és la IP de *nteum.remix.cat*, el servidor respondrà 192.168.0.1 (aquesta acció és coneguda com a *mapping*); de la mateixa manera, quan se li proporcioni l'adreça IP, respondrà amb el nom de la màquina (conegut com *reverse mapping*).

El *Domain Name System* (DNS) és una arquitectura arborescent que evita la duplicació de la informació i facilita la cerca. Per això, un únic DNS només té sentit com a part de l'arbre. Una de les aplicacions més utilitzades que presta aquest servei es diu *named*, s'inclou en la majoria de distribucions de GNU/Linux (`/usr/sbin/named`) i forma part d'un paquet anomenat `bind` (actualment versió 9.x), coordinat per l'ISC (Internet Software Consortium). El DNS és simplement una base de dades, per la qual cosa, és necessari que les persones que la modifiquin coneguin la seva estructura, ja que, en cas contrari, el servei quedarà afectat. Com a precaució, ha de tenir-se especial cura a guardar les còpies dels arxius per a evitar qualsevol interrupció en el servei. Els servidors DNS que poden convertir la majoria de nodes de DNS en la seva IP corresponent i es denominen *servidors DNS recursius*. Aquest tipus de

servidor no pot canviar els noms dels nodes de DNS que hi ha, simplement es pregunta a altres servidors DNS la IP d'un node DNS donat. Els servidors DNS autoritzats poden gestionar/canviar les adreces IP dels nodes DNS que gestionen i normalment són amb els quals contactarà un servidor DNS recursiu amb la finalitat de conèixer la IP d'un node DNS determinat. Una tercera variant dels servidors DNS són els DNS cau, que simplement emmagatzemen la informació obtinguda d'altres servidors DNS recursius.

1.1.1. Servidor de noms cau

En primer lloc, es configurarà un servidor de DNS per a resoldre consultes, que actuï com a cau per a les consultes de noms (*resolver, caching only server*). És a dir, la primera vegada consultarà el servidor adequat perquè es parteix d'una base de dades sense informació, però les vegades següents respondrà el servidor de noms cau, amb la corresponent disminució del temps de resposta. Per a instal·lar un servidor d'aquestes característiques, instal·lem els paquets `bind9` i `dnsutils` (p. ex., en Debian `apt-get bind9 dnsutils`).

Com es pot observar en el directori `/etc/bind`, trobarem una sèrie d'arxius de configuració, i entre aquests el principal és `named.conf`, que inclou altres `named.conf.options`, `named.conf.local`, `named.conf.default-zones` (podem trobar que aquesta configuració pot variar entre distribucions, però és similar). L'arxiu `/etc/bind/named.conf.options` conté les opcions generals per al servidor i els altres dos arxius ens serviran per a configurar les nostres zones en el servidor de noms que veurem en l'apartat següent. Per a configurar el nostre *caching only server*, hem de considerar que les preguntes en primer lloc les resoldrem nosaltres, però com que és evident que no tenim la resposta, s'haurà de redirigir la pregunta a servidors de DNS en la Xarxa. Per a això, és interessant considerar els de serveis com OpenDNS* que són les IP: 208.67.222.222 i 208.67.220.220 o bé serveis com els de Google** que responen a les IP: 8.8.8.8 i 8.8.4.4. A continuació, modifiquem l'arxiu `/etc/bind/named.conf.options` per a treure els comentaris de la secció *forwarders* posant el següent:

```
forwarders {
    // OpenDNS servers
    208.67.222.222;
    208.67.220.220;
    // Podríem incloure el nostre ISP/router -verificar la IP-
    192.168.1.1;
};
```

En el mateix arxiu al final i reemplaçant les que hi ha (es pot deixar aquesta configuració per a un segon pas), es podrien agregar opcions de seguretat perquè només resolgui les peticions de la nostra xarxa:

```
// Security options
listen-on port 53 { 127.0.0.1; 192.168.1.100; };
allow-query { 127.0.0.1; 192.168.1.0/24; };
```

*<http://www.opendns.com/opendns-ip-addresses/>
**<https://developers.google.com/speed/public-dns/?csw=1>

```
allow-recursion { 127.0.0.1; 192.168.1.0/24; };
allow-transfer { none; };
// Les següents dues opcions ja es troben per defecte en
// l'arxiu, però si no tenim IPv6 podem comentar l'última.
auth-nxdomain no; # conform to RFC1035
// listen-on-v6 { any; };
```

Es pot verificar la configuració amb `named-checkconf`. A continuació, modifiquem l'arxiu `/etc/resolv.conf` per a afegir `nameserver 127.0.0.1` perquè les preguntes a DNS la biblioteca `gethostbyname(3)` les faci al propi *host* i l'arxiu `/etc/nsswitch.conf` verificant que la línia `hosts` quedi com: `hosts: files dns` i indicarà l'ordre en què es resoldran les peticions (primer localment a `/etc/hosts` i després al servidor DNS). Finalment, només resta reiniciar el servidor amb `service bind9 restart` i fer les proves de funcionament. En el nostre cas, hem executat `dig www.debian.org` (s'han omès línies per a resumir la informació):

```
; «» DiG 9.8.4-rpz2+rl005.12-P1 «» www.debian.org
...
;; QUESTION SECTION:
;www.debian.org. IN A

;; ANSWER SECTION:
www.debian.org. 300 IN A 130.89.148.14
www.debian.org. 300 IN A 5.153.231.4
...
;; Query time: 213 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
...
```

On podem observar que la *query* ha trigat 213 mil·lisegons. Si la fem per segona vegada (ja ha quedat la consulta emmagatzemada en el nostre servidor cau):

```
; «» DiG 9.8.4-rpz2+rl005.12-P1 «» www.debian.org
...
;; QUESTION SECTION:
;www.debian.org. IN A

;; ANSWER SECTION:
www.debian.org. 293 IN A 5.153.231.4
www.debian.org. 293 IN A 130.89.148.14
...
;; Query time: 1 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
...
```

En aquest cas, la consulta ha trigat 1 mil·lisegon a verificar que el servidor cau ha funcionat. També podem fer la consulta inversa amb la instrucció `nslookup 130.89.148.14`, la qual cosa ens donarà el nom de la màquina que té assignada aquesta IP (en aquest cas, és el servidor al qual està assignat el domini `www.debian.org`):

```
Server: 127.0.0.1
Address: 127.0.0.1#53
Non-authoritative answer:
```

```
14.148.89.130.in-addr.arpa name = klecker4.snt.utwente.nl.  
Authoritative answers can be found from:  
 . nameserver = l.root-servers.net.  
 . nameserver = b.root-servers.net.  
 . nameserver = j.root-servers.net.  
 ...
```

És important considerar si es desitja instal·lar el servidor de DNS a una màquina virtual, i tenint en compte que és un servei al qual s'ha d'accedir des de fora, el servidor DNS (i la majoria de servidors) no pot estar en NAT sobre un *host* sinó que ha de tenir IP pròpia del segment de xarxa en la qual presta el servei i, per tant, l'adaptador de xarxa de la màquina virtual ha d'estar en configuració pont (*bridged*). Per a configurar els clients dins de la nostra xarxa, n'hi haurà prou de modificar l'arxiu */etc/resolv.conf* si són GNU/Linux (o a modificar */etc/network/interfaces* per a agregar-los com *dns-servers* IP si es té instal·lat el paquet *resolvconf*), i en Windows a modificar les propietats IPv4 de l'adaptador per a introduir l'IP del nostre servidor DNS cau.

1.1.2. Configuració d'un domini propi

El DNS posseeix una estructura en arbre i l'origen és conegut com a "." (vegeu */etc/bind/db.root*). Sota el "." hi ha els TLD (*Top Level Domains*) com **org**, **com**, **edu**, **net**, etc. Quan es busca en un servidor, si aquest no coneix la resposta, es buscarà de manera recursiva en l'arbre fins a trobar-la. Cada "." en una adreça (per exemple, *nteum.remix.cat*) indica una branca de l'arbre de DNS diferent i un àmbit de consulta (o de responsabilitat) diferent que s'anirà recorrent de manera recursiva d'esquerra a dreta.

Un altre aspecte important, a més del domini, és el *in-addr.arpa* (*inverse mapping*), el qual també està imbricat com els dominis i serveix per a obtenir noms quan es consulta per l'adreça IP. En aquest cas, les adreces s'escriuen a l'inrevés, en concordança amb el domini. Si *nteum.remix.world* és la 192.168.0.30, llavors s'escriurà com 30.0.168.192, en concordança amb *nteum.remix.world* per a la resolució inversa d'IP -> nom. En aquest exemple, utilitzarem un servidor de DNS que atén una xarxa interna (192.168.0.0/24) i després té una IP pública, però per a la finalitat de l'exemple i per a poder fer les proves internes la configurarem en una IP privada d'un altre segment (172.16.0.80) i un domini fictici (*remix.world*). En el cas de portar aquest servidor a producció, s'hauria de canviar per la IP externa que tingui i el domini corresponent que s'hagi obtingut i tinguem a la nostra disposició.

Per a configurar un servidor de noms propi, en primer lloc canviarem l'arxiu */etc/bind/named.conf* per a definir dues zones (una d'interna i una altra d'externa). L'arxiu quedarà com (observeu que hem comentat la línia que inclou *named.conf.default-zones*, ja que la inclourem després):

```
include "/etc/bind/named.conf.options";  
include "/etc/bind/named.conf.local";
```

```
//include "/etc/bind/named.conf.default-zones";
include "/etc/bind/named.conf.internal-zones";
include "/etc/bind/named.conf.external-zones";
```

Modifiquem l'arxiu *named.conf.internal-zones* per a indicar-li els arxius on es trobaran realment els noms de les màquines, que en aquest cas seran dos: un per a resolució nom -> IP anomenat *remix.world.lan* i un altre per a la resolució inversa IP -> nom anomenat *0.168.192.db*.

```
view "internal" {
    match-clients {
        localhost;
        192.168.0.0/24;
    };
    // set zone for internal
    zone "remix.world" {
        type master;
        file "/etc/bind/remix.world.lan";
        allow-update { none; };
    };
    // set zone for internal reverse
    zone "0.168.192.in-addr.arpa" {
        type master;
        file "/etc/bind/0.168.192.db";
        allow-update { none; };
    };
    include "/etc/bind/named.conf.default-zones";
};
```

De la mateixa manera, modifiquem el fitxer *named.conf.external-zones* per a indicar-li els arxius que resoldran aquesta zona i que també seran dos: *remix.world.wan* i *80.0.16.172.db* per a la resolució directa i inversa, respectivament.

```
view "external" {
    // define for external section
    match-clients { any; };
    // allow any query
    allow-query { any; };
    // prohibit recursion
    recursion no;
    // set zone for external
    zone "remix.world" {
        type master;
        file "/etc/bind/remix.world.wan";
        allow-update { none; };
    };
    // set zone for external reverse
    zone "80.0.16.172.in-addr.arpa" {
        type master;
        file "/etc/bind/80.0.16.172.db";
        allow-update { none; };
    };
};
```

També s'ha de modificar el *named.conf.options*:

```
options {
    directory "/var/cache/bind";
    // ...
    // forwarders {
    // 158.109.0.9;
```



```
// 158.109.0.1;
// };
// query range you permit
// allow-query { localhost; 192.168.0.0/24; };
// the range to transfer zone files
// allow-transfer { localhost; 192.168.0.0/24; };
// recursion range you allow
// allow-recursion { localhost; 192.168.0.0/24; };

dnssec-validation auto;
auth-nxdomain no; # conform to RFC1035
//listen-on-v6 { any; };
};
```

Ara és necessari definir els arxius que realment resoldran les IP/nom de les zones. Comencem per *remix.server.lan*:

```
$TTL 86400
@ IN SOA nteum.remix.world. root.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

IN NS nteum.remix.world.
IN A 192.168.0.30
IN MX 10 nteum.remix.world.

nteum IN A 192.168.0.30
```

S'ha de tenir en compte el “.” al final dels noms de domini. El significat dels registres indiquen el següent: **SOA** (*Start of Authority*) ha d'estar en tots els arxius de zona a l'inici, després de **TTL** (*Time To Live*), que indica el temps en segons que els recursos d'una zona seran vàlids, el símbol **@** significa l'origen del domini; **NS**, el servidor de noms per al domini, **MX** (*Mail eXchange*) indica on s'ha de dirigir el correu per a una determinada zona, i **A** és l'IP que s'ha d'assignar a un nom. Per a l'arxiu de resolució inversa *0.168.192.db*:

```
$TTL 86400
@ IN SOA nteum.remix.world. nteum.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

IN NS nteum.remix.world.
IN PTR remix.world.
IN A 255.255.255.0
30 IN PTR nteum.remix.world.
```

On podem observar que els registres tenen un format <últim-dígit-IP> IN <PTR FQDN-of-system> on PTR (*Registre PoinTeR*) crea un apuntador cap al nom de l'IP que coincideix amb el registre. De la mateixa manera procedim per a la zona externa amb l'arxiu *remix.world.wan*:

```
$TTL 86400
@ IN SOA nteum.remix.world. root.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

IN NS nteum.remix.world.
IN A 172.16.0.82
IN MX 10 nteum.remix.world.
nteum IN A 172.16.0.82
```

I la seva resolució inversa *80.0.16.172.db*:

```
$TTL 86400
@ IN SOA nteum.remix.world. nteum.remix.world. (
    2013050601 ;Serial
    3600 ;Refresh
    1800 ;Retry
    604800 ;Expire
    86400 ;Minimum TTL
)

IN NS nteum.remix.world.
IN PTR remix.world.
IN A 255.255.255.248
82 IN PTR nteum.remix.world.
```

Ara només restaria canviar el fitxer */etc/resolv.conf* per a incloure `nameserver 192.168.0.30` i reiniciar el servidor service `bind9 restart`. S'ha de mirar */var/log/syslog* per a observar si hi ha errors en l'arrencada del servidor i corregir-los fins que el seu reinici estigui lliure d'aquests (els més comuns són els símbols utilitzats com a comentaris, els punts al final de domini o espais abans dels registres A-PTR). Després, podem provar amb la instrucció `dig nteum.remix.world` i tindrem una sortida com:

```
; «» DiG 9.8.4-rpz2+rl005.12-P1 «» nteum.remix.world
...
;; QUESTION SECTION:
;nteum.remix.world. IN A
;; ANSWER SECTION:
nteum.remix.world. 86400 IN A 192.168.0.30
;; AUTHORITY SECTION:
remix.world. 86400 IN NS nteum.remix.world.
;; Query time: 0 msec
...
```

Si fem la petició inversa amb `dig -x 192.168.0.30`, tindrem:

```
...
;; QUESTION SECTION:
;30.0.168.192.in-addr.arpa. IN PTR
;; ANSWER SECTION:
30.0.168.192.in-addr.arpa. 86400 IN PTR nteum.remix.world.
;; AUTHORITY SECTION:
0.168.192.in-addr.arpa. 86400 IN NS nteum.remix.world.
;; ADDITIONAL SECTION:
nteum.remix.world. 86400 IN A 192.168.0.30
...
```

Un aspecte molt interessant és, per exemple, posar àlies, la qual cosa significa incloure un registre CNAME en *remix.world.lan* com `ftp IN CNAME nteum.remix.world`. Atès que és un servei essencial, és necessari moltes vegades disposar d'un servei secundari (*slave*), i *bind9* permet crear molt fàcilment serveis d'aquest tipus a partir del servidor primari per transferència de les taules (http://www.server-world.info/en/note?os=debian_7.0&p=dns&f=5).

1.1.3. Altres DNS

MaraDNS és un servidor DNS que pot funcionar com a DNS recursiu (*cau*) o com a *authoritative name server* i que es caracteritza per la seva extremadament fàcil configuració. MaraDNS ha estat optimitzat per a un petit nombre de dominis de manera simple i eficient i que permet crear un domini propi i servir-lo sense grans esforços, encara que conté tots els elements per a configurar-lo de manera similar a *bind9*. En el seu disseny, s'ha posat especial cura en la seguretat i utilitza una biblioteca de gestió de cadenes de caràcters que resisteixen els principals atacs als DNS per desbordament (*buffer overflows*)[6]. El paquet MaraDNS inclou un DNS autoritzat (*maradns*), un servidor DNS recursiu amb possibilitats de *cau* i que es diu *Deadwood*. La decisió de posar un servidor autoritzat o recursiu dependrà de la funció que es busqui. Si simplement és per a obtenir altres llocs a Internet, s'haurà de configurar un servidor recursiu. En canvi, si es desitja registrar dominis i es té una xarxa d'ordinadors dins d'aquest domini, cal configurar un servidor DNS amb autoritat.

Com a proves d'aquest servei, configurarem diferents opcions de MaraDNS per a Debian. Si bé el paquet està inclòs en el repositori, és una versió que té problemes de seguretat (<http://maradns.samiam.org/debian.html>) i el seu autor recomana baixar el codi font i compilar-lo/instal·lar-lo. Per a això, cal obtenir el fitxer des de l'adreça <http://maradns.samiam.org/download.html>, descomprimir-lo (`tar xjvf maradns-x.x.xx.tar.bz2` on *x.x.xx* és la versió del programari descarregat) i dins del directori executar `./configure; make` i si no hi ha errors executar `make install`. Els servidors s'instal·laran en `/usr/local/sbin` i altres arxius complementaris en `/usr/local/bin`. Els arxius de configuració en `/etc/mararc` i `/etc/dwood3rc` i el directori de configuració `/etc/maradns`.

En primer lloc, per a configurar un DNS recursiu i *cau*, s'ha de modificar l'arxiu `/etc/dwood3rc` per a incloure:

```
bind_address="127.0.0.1"
chroot_dir = "/etc/maradns"
#upstream_servers = {}
#upstream_servers["."] = "8.8.8.8, 8.8.4.4"
recursive_acl = "127.0.0.1/16" # Qui pot fer servir la cau

# Paràmetres de configuració del servidor
maxprocs = 8 # Maximum number of pending requests
handle_overload = 1 # Send SERVER FAIL when overloaded
maradns_uid = 99 # UID Deadwood runs as
maradns_gid = 99 # GID Deadwood runs as
maximum_cache_elements = 60000
```

```
# File cache (readable and writable by the maradns_uid/gid)
cache_file = "dw_cache"
```

També és possible que Deadwood pugui contactar amb altres DNS recursius abans que amb els *root DNS servers*. Per a això, s'ha de treure el comentari de les línies *upstream_servers* (en aquest cas, s'hi han posat els DNS públics de Google). Després, podrem executar `/usr/local/sbin/Deadwood`, o si es vol executar com a *daemon*, s'haurà d'utilitzar el programa *duende* inclòs en el paquet: `/usr/local/bin/duende /usr/local/sbin/Deadwood`.

Per a configurar un DNS autoritzat, hem de modificar l'arxiu `/etc/mararc` amb els següents valors: *cvs2*, el domini que gestionarà aquest servidor i l'arxiu que tindrà la informació en aquest cas *db.local*, el directori on es trobaran els arxius de configuració (`/etc/maradns`) i l'IP d'on respondrà.

```
cvs2 = {}
cvs2["remix.world."] = "db.local"
ipv4_bind_addresses = "127.0.0.1"
chroot_dir = "/etc/maradns"
```

L'arxiu `/etc/maradns/db.local` només tindrà una línia per cada registre que desitgem que transformi el DNS, per exemple,

```
nteum.remix.world. 192.168.0.30 ~
```

A continuació s'ha d'engegar el servidor (`/usr/local/sbin/maradns`), o si es desitja executar com a *daemon*, aleshores s'haurà d'utilitzar el programa `/usr/local/bin/duende /usr/local/sbin/maradns`. Una de les qüestions interessants és la següent: com podem donar servei de manera *authoritative* per als nostres dominis en una Intranet però, a més, ser recursius quan la petició sigui a dominis externs? És a dir, tenim uns pocs noms (FQDN) que pertanyen al nostre domini (per exemple, *remix.world*) sobre la nostre Intranet i desitgem al mateix temps assegurar la resolució de domini FQDN externs (per exemple, *debian.org*). La solució plantejada per l'autor (S. Trenholme*) passa per una configuració combinada de MaraDNS 2.x i Deadwood 3.x posant MaraDNS que escolti sobre localhost (127.0.0.1) mentre que Deadwood ho fa sobre l'IP del servidor (192.168.1.37, en el nostre cas). Els arxius de configuració `/etc/mararc` i `/etc/dwood3rc` seran:

```
# Arxiu /etc/mararc
# Simplement, indicar que escolti sobre localhost i el domini que serà responsable
ipv4_bind_addresses = "127.0.0.1"
timestamp_type = 2
verbose_level = 3
hide_disclaimer = "YES"
cvs2 = {}
cvs2["remix.world."] = "db.local"
chroot_dir = "/etc/maradns"

#Arquivo /etc/dwood3rc
# Indicar-li que a més del nostre domini on volem consultar la resta
root_servers = {}
root_servers["remix.world."] = "127.0.0.1"
```

*<http://woodlane.webconquest.com/pipermail/list/2011-august/000928.html>

```
root_servers["."]="198.41.0.4, 192.228.79.201, 192.33.4.12, 199.7.91.13,"
root_servers["."]+= "192.203.230.10, 192.5.5.241, 192.112.36.4, 128.63.2.53, "
root_servers["."]+= "192.36.148.17, 192.58.128.30, 193.0.14.129, 199.7.83.42, "
root_servers["."]+= "202.12.27.33"
# On escoltarà les peticions = IP del servidor Deadwood
bind_address="192.168.1.37"
# Habilitar la resposta d'IP privades
filter_rfc1918=0
# IP que podran fer peticions
recursive_acl = "192.168.1.0/24"
# Cau de disc per a Deadwood
cache_file = "dw_cache"
# Directori arrel
chroot_dir = "/etc/maradns"
```

Finalment, en les nostres màquines posarem com a */etc/resolv.conf* l'IP del nostre servidor `nameserver 192.168.1.37` i engegarem tots dos servidors (després de verificar que funciona, es poden posar com a *daemons* i baixar el nivell de *log* amb `verbose_level = 1`).

Un altre servidor interessant que ve en moltes distribucions és **Dnsmasq**, que permet de manera simple configurar un servidor DNS (*forwarder*) i un servidor DHCP en el mateix paquet per a xarxes petites/mitjanes. És possible atendre peticions de noms de màquines que no estan en els DNS globals i integrar el servidor de DHCP per a permetre que màquines gestionades pel DHCP apareguin en el DNS amb noms ja configurats, i a més suporta gestió d'IP dinàmiques i estàtiques pel DHCP i permet l'accés BOOTP/TFTP per a màquines sense disc i que fan la seva inicialització per xarxa.

La forma de configurar ràpidament un DNS per a màquines de domini propi i que faci de *forwarder* per als dominis externs és `apt-get update; apt-get install dnsmasq`. Si el que es desitja és un simple DNS, ja està configurat tenint en compte que en */etc/resolv.conf* tindrem alguna cosa com `nameserver 8.8.8.8`. Amb això, podem provar els externs utilitzant la instrucció `dig debian.org @localhost` (o simplement `dig debian.org`), però també respondrà a totes les màquines que tinguem definides en */etc/hosts*. Per exemple, si tenim una línia com `192.168.1.37 nteum.remix.world nteum` podrem executar `dig nteum @localhost` i ens respondrà amb l'IP corresponent. Per aquest motiu, si la xarxa és simple podem agregar les màquines en aquest arxiu, però si és més complexa, podem utilitzar l'arxiu de configuració */etc/dnsmasq.conf* que inclou una gran quantitat d'opcions per a organitzar els dominis interns i altres paràmetres no només per a la configuració del DNS, sinó també per al servidor de DHCP.[7]

1.2. NIS (YP)

Amb la finalitat de facilitar l'administració i donar comoditat a l'usuari en xarxes de diferents mides que executen GNU/Linux (o algun altre sistema operatiu amb suport per a aquest servei), s'executen serveis de *Network Information Service*, NIS (o *Yellow Pages*, YP, en la definició original de Sun). GNU/Linux pot donar suport com a client/servidor de NIS. La informació

que es pot distribuir en NIS és aquesta: usuaris (*login names*), contrasenyes (*passwords*, */etc/passwd*), directoris d'usuari (*home directories*) i informació de grups (*group information*, */etc/group*), xarxes, *hosts*, etc. Això presenta l'avantatge que, des de qualsevol màquina client o des del mateix servidor, l'usuari es podrà connectar amb el mateix compte i contrasenya i en el mateix directori (encara que el directori haurà de ser muntat anteriorment sobre totes les màquines client per NFS o mitjançant el servei de *automount*). [10, 11]

L'arquitectura NIS és del tipus client-servidor, és a dir, hi ha un servidor que disposarà de totes les bases de dades i uns clients que consultaran aquestes dades de manera transparent per a l'usuari. Per això, s'ha de pensar en la possibilitat de configurar servidors "de reforç" (anomenats *secundaris*) perquè els usuaris no quedin bloquejats davant la caiguda del servidor principal. Per aquest motiu, l'arquitectura es denomina realment *de múltiples servidors* (*master+mirrors-clients*).

1.2.1. Com es pot iniciar un client local de NIS en Debian?

Un client local és el que annexa l'ordinador a un domini NIS ja existent. Primer s'ha de verificar que es tenen instal·lats els paquets *netbase* (xarxa bàsica TCP/IP) i *rpcbind* (servidor que converteix nombres RPC –*Remote Procedure Call*– en ports DARPA) i *nis* (específic). És important destacar que el paquet *rpcbind*, que ja està en la majoria de distribucions, és un servei que substitueix el tradicional *portmap*. Aquest ha quedat obsolet per diferents causes vinculades als canvis en el kernel de NFSV3/V4, però especialment, perquè no té suport per a IPv6. És necessari verificar la instal·lació dels dos primers paquets per a tots els programes que executen RPC, incloent-hi NFS i NIS. Es recomana fer servir l'ordre *apt-get* (o també *dpkg*, *aptitude*, *synaptic*), i es pot verificar si està instal·lat amb la instrucció *apt-cache pkgnames* en mode text.

El procediment per a la instal·lació del paquet NIS sol·licitarà un domini (NIS *domainname*). Aquest és un nom que descriurà el conjunt de màquines que utilitzaran el NIS (no és un nom de *host*). Cal tenir en compte que NISNteum és diferent de Nisnteum com a nom de domini. Per a configurarlo, es pot utilitzar l'ordre *nisdomainname*, domini que s'emmagatzemarà en */proc/sys/kernel/domainname*. També es pot reconfigurar el paquet amb *dpkg-reconfigure nis* i tornarem a començar, podem veure que el *rpcbind* està en marxa amb *ps -ef | grep rpcbind* o també amb *rpcinfo -p* i ens mostrarà diferents línies (per a diferents programes, ports/protocols, versions) com a *portmapper*. Per a reiniciar el *rpcbind*, podem fer *service rpcbind restart*. És important tenir en compte que la primera vegada que instal·lem en NIS, que serà el client en el nostre cas, el sistema d'instal·lació intentarà posar els *daemons* en marxa però fallarà, ja que no tenim cap servidor configurat (que serà el següent pas) i donarà un missatge de [...] *Starting NIS services: ypbind[...] binding to YP server...failed (backgrounded)*.

El client NIS utilitza l'ordre `ypbind` per a trobar un servidor per al domini especificat, mitjançant `broadcast` (no aconsellat per insegur) o buscant el servidor indicat en l'arxiu de configuració `/etc/yp.conf` (recomanable), que pot tenir bàsicament dos tipus de configuració com les mostrades a sota –només s'ha d'indicar una d'aquestes, encara que és recomanable la primera.

Sintaxi de l'arxiu `/etc/yp.conf`

`domain nisdomain server hostname:` indica que s'utilitza el *hostname* per al domini *nisdomain*. En el nostre cas, podria ser `domain NISnteum server 192.168.1.30`. Es podria tenir més d'una entrada d'aquest tipus per a un únic domini.

`ypserver hostname:` indica que s'utilitza *hostname* com a servidor introduint l'adreça IP del servidor NIS. Si s'indica el nom, ens hem d'assegurar que es pot trobar la IP per DNS o que la mateixa figura en l'arxiu `/etc/hosts`, ja que, d'una altra manera, el client es bloquejarà.

Per a iniciar el servei, s'ha d'executar:

```
/etc/init.d/nis stop      per a aturar-lo
/etc/init.d/nis start     per a iniciar-lo
o també amb l'ordre service nis start|stop
```

A partir d'aquest moment, el client NIS estarà funcionant. Això es pot confirmar amb `rpcinfo -o localhost ypbind`, que mostrarà les dues versions del protocol actiu. Quan el servidor estigui funcionant, es pot utilitzar l'ordre `ypcat mapname` (per exemple, `ypcat passwd`, que mostrarà els usuaris NIS definits en el servidor), en què les relacions entre *mapnames* i les taules de la base de dades NIS estan definides en `/var/yp/nicknames`.

1.2.2. Quins recursos s'han d'especificar per a utilitzar el NIS?

A partir de la inclusió de `lib6` en les distribucions de GNU/Linux (això és Debian5 o FC4), és necessari orientar la consulta del *login* a les bases de dades adequades amb els passos següents:

1) Verificar el fitxer `/etc/nsswitch.conf` i assegurar-se que les entrades `passwd`, `group`, `shadow` i `netgroup` són similars a:

```
passwd: compat nis
group: compat nis
shadow: compat nis
netgroup: nis
hosts: files dns nis
```

2) Consulteu la sintaxi d'aquest arxiu en `man nsswitch.conf`.

3) En determinades configuracions, pot ser necessari agregar al final del fitxer */etc/pam.d/common-session* (si no existeix la línia)

```
session optional pam_mkhomedir.so skel=/etc/skel umask=077
```

per a crear automàticament el directori *home* si no existeix, però no és l'habitual, ja que generalment es desitja que aquest es munti del servidor NFS.

4) Amb aquesta configuració, es podrà fer una connexió local (sobre el client NIS) a un usuari que no estigui definit en el fitxer */etc/passwd*, és a dir, un usuari definit en una altra màquina (*ypserver*). Per exemple, es podria fer `ssh -l user localhost`, en què *user* és un usuari definit en *ypserver*.

1.2.3. Com s'ha de configurar un servidor?

Abans d'instal·lar el servidor, cal tenir instal·lat el paquet `nis i rpcbind` sobre la màquina que farà de servidor (`apt-get install nis rpcbind`) i configurar el domini -el mateix que hem introduït quan hem configurat el client (NISnteu en el nostre cas). Després, s'ha de modificar l'arxiu */etc/default/nis* per a modificar la línia `NISSERVER=master` per a indicar-li el rol de servidor. També (encara que es pot fer en un segon pas) s'ha d'ajustar l'arxiu */etc/ypserv.securenets* per a modificar la línia que indica `0.0.0.0 0.0.0.0` que dóna accés a tothom i restringir-lo a la nostra xarxa, p. ex. `255.255.255.0 192.168.1.0`. Finalment, s'ha de modificar el */etc/hosts* perquè tinguem la màquina amb el nom definit en */etc/hostname* (es pot canviar/visualitzar amb l'ordre `hostname` o editant el fitxer) amb una línia FQND com, per exemple, `192.168.1.30 nteum.remix.world nteum`. La configuració del servidor es porta a terme amb l'ordre `/usr/lib/yp/ypinit -m`; en algunes distribucions, cal verificar que existeix l'arxiu */etc/networks*, que és imprescindible per a aquest *script*. Si aquest arxiu no existeix, n'heu de crear un de buit amb `touch/etc/networks`; aquest *script* ens sol·licitarà quins seran els servidors NIS indicant-nos la màquina local per defecte, i haurèm d'acabar amb `Ctrl+D`. També es pot executar el client `ypbind` sobre el servidor; així, tots els usuaris entren per NIS indicant en l'arxiu */etc/default/nis* que existeixi `NISCLIENT=true`.

Considereu que, a partir d'aquest moment, les ordres per a canviar la contrasenya o la informació dels usuaris, com `passwd`, `chfn` o `adduser`, no són vàlides. En el seu lloc, s'hauran d'utilitzar ordres com ara `yppasswd`, `ypchsh` i `ypchfn`. Si es canvien els usuaris o es modifiquen els arxius esmentats, caldrà reconstruir les taules de NIS executant l'ordre `make` en el directori */var/yp* per a actualitzar les taules.

La configuració d'un servidor esclau és similar a la del mestre, excepte si `NISSERVER = slave` en */etc/default/nis*. Sobre el mestre, s'ha d'indicar que distribueixi les taules automàticament als esclaus, posant `NOPUSH = "false"` en l'arxiu */var/yp/Makefile*. Ara s'ha d'indicar al mestre qui és el seu esclau mitjançant l'execució de:


```
/usr/lib/yp/ypinit -m
```

i introduint els noms dels servidors esclaus. Això reconstruirà els mapes, però no enviarà els arxius als esclaus. Per a això, sobre l'esclau, executeu:

```
/etc/init.d/nis stop
/etc/init.d/nis start
/usr/lib/yp/ypinit -s nombre_master_server
```

D'aquesta manera, l'esclau carregarà les taules des del mestre. També es podria posar en el directori `/etc/cron.d` l'arxiu NIS amb un contingut similar a (recordeu fer un `chmod 755 /etc/cron.d/nis`):

```
20 * * * * root /usr/lib/yp/ypxfr_1perhour >/dev/null 2>&1
40 6 * * * root /usr/lib/yp/ypxfr_1perday >/dev/null 2>&1
55 6,18 * * * root /usr/lib/yp/ypxfr_2perday >/dev/null 2>&1
```

Amb això, es garantirà que tots els canvis del mestre es transfereixin al servidor NIS esclau.

Inicieu el servidor executant primer l'ordre `/etc/init.d/nis stop` i després la instrucció `/etc/init.d/nis start`. Aquesta sentència iniciarà el servidor (`ypserv`) i el *password daemon* (`yppasswdd`), l'activació del qual es podrà consultar amb `ypwich -d domain`.

Actualització de les taules NIS

És recomanable que després de fer servir `adduser` o `useradd` per a agregar un nou usuari sobre el servidor, executeu `cd /var/yp; make` per a actualitzar les taules NIS (i sempre que es canviï alguna característica de l'usuari, per exemple la paraula clau amb l'ordre `passwd`, només canviarà la contrasenya local i no la del NIS).

Per a provar que el sistema està funcionant i que l'usuari donat d'alta està en el NIS, podeu fer `ypmatch userid passwd`, on `userid` és l'usuari donat d'alta amb `adduser` abans i després d'haver fet el `make`. Per a verificar el funcionament del sistema NIS, podeu utilitzar l'*script* de <http://tldp.org/howto/nis-howto/verification.html> que permet una verificació més detallada del NIS.

També es pot provar en la mateixa màquina que un usuari es pot connectar a un domini NIS fent: `adduser usuari` i responent les preguntes, després `cd /var/yp; make`. En aquest moment, ho veurem tant en `/etc/passwd` com en NIS amb `ypcat passwd`. Després, ho podem esborrar amb l'ordre `userdel usuari` (no s'ha de fer servir l'ordre `deluser` sinó el `userdel`), que només ho esborrarà del `/etc/passwd` i no del NIS (cal anar amb compte de no fer un `make`, ja que aquest també ho esborrarà del NIS). Després ens podem connectar com a `ssh usuario@localhost`, i amb això podrem verificar que el NIS funciona, ja que l'accés només es podrà fer per NIS perquè per a `/etc/passwd` no existeix aquest usuari. En relació amb NIS+, el seu desenvolupament va ser aturat el 2012 a causa de la falta d'interès/recursos de la comunitat, no així NIS/YP, que continua actualitzat i present en totes les distribucions (<http://www.linux-nis.org/nisplus/>).

1.3. Serveis de connexió remota: Telnet i ssh

1.3.1. Telnet i telnetd

Telnet és una ordre (client) utilitzat per a comunicar-se de manera interactiva amb un altre *host* que executa el *daemon* `telnetd`. L'ordre `telnet` es pot executar com `telnet host` o interactivament com `telnet`, el qual posarà el *prompt* “telnet>” i després, per exemple, `open host`. Una vegada establerta la comunicació, s’haurà d’introduir l’usuari i la contrasenya sota la qual es desitja connectar al sistema remot. Es disposa de diverses ordres (en mode interactiva) com `open`, `logout` i `mode` (s’han de definir les característiques de visualització), `close`, `encrypt`, `quit`, `set` i `unset` o es poden executar ordres externes amb “!”. Es pot utilitzar l’arxiu `/etc/telnetrc` per a definicions per defecte o `.telnetrc` per a definicions d’un usuari particular (haurà d’estar en el directori *home* de l’usuari).

El *daemon* `telnetd` és el servidor de protocol Telnet per a la connexió interactiva. Generalment, és el *daemon* `inetd` que engega `telnetd`; es recomana incloure un *wrapper* `tcpd` (que utilitza les regles d’accés en `host.allow` i `host.deny`) en la crida al `telnetd` dins de l’arxiu `/etc/inetd.conf`. Per a incrementar la seguretat del sistema s’hi hauria de incloure, per exemple, una línia com:

```
telnet stream tcp nowait telnetd.telenetd /usr/sbin/tcpd /usr/bin/in.telnetd
```

En les distribucions actuals, la funcionalitat de `inetd` s’ha reemplaçat per la de `xinetd`, que requereix la configuració de l’arxiu `/etc/xinetd.conf` que s’ha explicat en l’apartat de configuració de xarxa de l’assignatura *Administració GNU/Linux*. Si es vol engegar `inetd` a mode de proves, es pot fer servir la sentència `/etc/init.d/inetd.real start`. Si l’arxiu `/etc/uissue.net` està present, el `telnetd` mostrarà el seu contingut a l’inici de la sessió. També es pot usar `/etc/security/access.conf` per a habilitar i deshabilitar *logins* d’usuari, *host* o grups d’usuaris, segons es connectin.

S’ha de recordar que, si bé el parell `telnet-telnetd` pot funcionar en mode *encrypt* en les últimes versions (transferència de dades encriptades, que han d’estar compilades amb l’opció corresponent), és una ordre que ha quedat en l’oblit per la seva falta de seguretat (transmet el text net sobre la xarxa, la qual cosa permet la visualització del contingut de la comunicació entre dues màquines, per exemple, amb l’ordre `tcpdump`); no obstant això, es pot fer servir en xarxes segures o situacions controlades.

Si no està instal·lat, es pot utilitzar (Debian) `apt-get install telnetd` i després verificar que s’ha donat d’alta, o bé en `/etc/inetd.conf`, o bé en `/etc/xinetd.conf` (o en el directori on estiguin definits els arxius; per exemple, `/etc/xinetd.d` segons s’indiqui en l’arxiu anterior amb la sentència

include /etc/xinetd.d). L'arxiu `xinetd.conf` o l'arxiu `/etc/xinetd.d/telnetd` hauran d'incloure una secció com*:

```
service telnet {
    disable = no
    flags = REUSE
    socket_type = stream
    wait = no
    user = root
    server = /usr/sbin/in.telnetd
    log_on_failure += USERID
}
```

*Qualsevol modificació en `xinetd.conf` haurà d'arrencar novament el servei amb `service xinetd restart`.

SSL telnet(d)

Es recomana que en lloc de fer servir `telnetd` s'utilitzi `SSL telnet(d)`, que reemplaça a `telnet(d)` utilitzant encriptació i autenticació per SSL, o que s'utilitzi SSH (el qual ja és un estàndard en totes les distribucions i entorns). L'`SSLTelnet(d)` pot funcionar amb el `telnet(d)` normal en ambdues adreces ja que a l'inici de la comunicació verifica si l'altre costat (*peer*) suporta SSL, i en cas contrari continua amb el protocol `telnet` normal. Els avantatges pel que fa a `SSL telnet(d)` són que les seves contrasenyes i dades no circularan per la xarxa en mode de text net i ningú que utilitzi l'ordre abans esmentada (`tcpdump`) podrà veure el contingut de la comunicació. També `SSL telnet` es pot utilitzar per a connectar-se, per exemple, a un servidor web segur (per exemple `https://servidor.web.org`) simplement fent `telnet servidor.web.org 443`.

1.3.2. SSH, *Secure shell*

Un canvi aconsellable avui dia és utilitzar `ssh` (ordre que ja està en tots els sistemes operatius, també en Windows, amb `putty`* o `moabterm`***) en lloc de `telnet`, `rlogin` o `rsh`. Aquestes tres últimes ordres són insegures (excepte `SSLTelnet`) per diverses raons. La més important és que tot el que es transmet per la xarxa, inclosos noms d'usuaris i contrasenyes, és en text net (encara que hi ha versions de `telnet-telnetd` encriptats, ha de coincidir que tots dos ho siguin), de manera que qualsevol que tingui accés a aquesta xarxa o a algun segment d'aquesta pot obtenir tota aquesta informació i després suplantar la identitat de l'usuari. La segona és que aquests ports (`telnet`, `rsh`, etc.) són el primer lloc on un *cracker* intentarà connectar-se. El protocol `ssh` (en la seva versió OpenSSH) proporciona una connexió encriptada i comprimida molt més segura que, per exemple, `telnet` (és recomanable utilitzar la versió 2.0 o versions superiors del protocol). Totes les distribucions actuals incorporen el client `ssh` i el servidor `sshd` per defecte. És necessari tenir actualitzada la biblioteca OpenSSL, utilitzada per aquests programes, ja que recentment es va trobar un problema de seguretat anomenat *heartbleed* i que ha estat ràpidament solucionat en la majoria de distribucions GNU/Linux (<http://www.kriptopolis.com/recomendaciones-heartbleed>).

*<http://www.putty.org>
 **<http://mobaxterm.mobatek.net>

ssh

Per a executar l'ordre, feu:

```
ssh -l login-name host o ssh user@hostname
```

Mitjançant SSH, es poden encapsular altres connexions com X11 o qualsevol altra TCP/IP. Si s'omet el paràmetre `-l`, l'usuari es connectarà amb el mateix usuari local i en tots dos casos el servidor sol·licitarà la contrasenya per a validar la identitat de l'usuari. SSH suporta diferents modes d'autenticació (vegeu `man ssh`) basades en algorisme RSA i clau pública. Si el client no està instal·lat, cal fer `apt-get install openssh-client`.

Usant l'ordre `ssh-keygen -t rsa|dsa`, es poden crear les claus d'identificació d'usuari. L'ordre crea en el directori del `.ssh` de l'usuari els fitxers `*id_rsa` i `id_rsa.pub`, les claus privada i pública, respectivament. L'usuari podria copiar la clau pública (`id_rsa.pub`) en l'arxiu `$HOME/.ssh/authorized_keys` del directori `.ssh` de l'usuari de la màquina remota. Aquest arxiu podrà contenir tantes claus públiques com llocs des d'on es vulgui connectar a aquesta màquina de manera remota. La sintaxi és d'una clau per línia, encara que les línies tindran una grandària considerable. Després d'haver introduït les claus públiques de l'usuari-màquina en aquest arxiu, aquest usuari es podrà connectar sense contrasenya des d'aquesta màquina. També per a efectuar aquesta còpia, es pot utilitzar l'ordre `ssh-copy-id màquina`, que copiarà les claus de manera automàtica i és molt més simple d'utilitzar.

*Per exemple, per a l'algorisme d'encryptació RSA.

De manera normal (si no s'han creat les claus), es demanarà a l'usuari una contrasenya, però com que la comunicació serà sempre encriptada, mai serà accessible a altres usuaris que puguin escoltar sobre la Xarxa. Per a major informació, consulteu `man ssh`. Per a executar remotament una ordre, simplement feu:

```
ssh -l login name host_ordre_remot  
Per exemple: ssh -l user localhost ls -al
```

sshd

El `sshd` és el servidor (*daemon*) per al `ssh` (si no estan instal·lats, es pot fer amb `apt-get install openssh-client openssh-server`. Junts reemplacen `rlogin`, `telnet`, `rsh` i proveeixen una comunicació segura i encriptada entre dos *hosts* insegurs de la Xarxa. Aquest s'arrenca generalment mitjançant els arxius d'inicialització (`/etc/init.d` o `/etc/rc`) i espera connexions dels clients. El `sshd` de la majoria de distribucions actuals suporta les versions 1, 2 i 3 del protocol SSH. Quan s'instal·la el paquet, es crea una clau RSA específica del *host* i quan el *daemon* s'inicia, en crea una altra, l'RSA per a la sessió, que no s'emmagatzema en el disc i que canvia cada hora. Quan un client inicia la comunicació, genera un nombre aleatori de 256 bits que està encriptat amb les dues claus del servidor i és enviat. Aquest nombre s'utilitzarà durant la comunicació com a clau de sessió per a encriptar la comunicació que es farà per mitjà d'un algorisme d'encryptació estàndard. L'usuari pot seleccionar qualsevol dels algorismes disponibles oferts pel servidor. Hi ha algunes diferències (més

segur) quan s'utilitza la versió 2 (o 3) del protocol. A partir d'aquest moment, s'inicien alguns dels mètodes d'autenticació d'usuari descrits en el client o se li sol·licita la contrasenya, però sempre amb la comunicació encriptada. Per a més informació, consulteu `man sshd`.

Tant `ssh` com `sshd` tenen un gran conjunt de paràmetres que poden ser configurats segons es necessiti en `/etc/ssh/ssh_config` i `/etc/ssh/sshd_config`. En el servidor, probablement algunes de les opcions més usades són `PermitRootLogin yes|no` per a permetre la connexió de l'usuari `root` o no, `IgnoreRhosts yes` per a evitar llegir els arxius `$HOME/.rhosts` i `$HOME/.shosts` dels usuaris, `X11Forwarding yes` per a permetre que aplicacions Xwindows en el servidor es visualitzin a la pantalla del client (molt útil en l'administració remota de servidors amb l'ordre `ssh -X host_per_a_administrar`).

Túnel sobre SSH

Moltes vegades tenim accés a un servidor `sshd`, però per qüestions de seguretat no podem accedir a altres serveis que no estan encriptats (per exemple, un servei de consulta de correu POP3 o un servidor de finestres X11) o simplement es vol connectar amb un servei al qual només es té accés des de l'entorn de l'empresa. Per a això, és possible establir un túnel encriptat entre la màquina client (per exemple, amb Windows i un client `ssh` anomenat `putty` de programari lliure) i el servidor amb `sshd`. En aquest cas, en vincular el túnel amb el servei, el servei veurà la petició com si vingués de la mateixa màquina. Per exemple, si volem establir una connexió per a POP3 sobre el port 110 de la màquina remota (i que també té un servidor `sshd`), farem:

```
ssh -C -L 1100:localhost:110 usuario-id@host
```

Aquesta ordre demanarà la contrasenya per a l'usuari-id sobre *host*, i una vegada connectat, s'haurà creat el túnel. Cada paquet que s'envii des de la màquina local sobre el port 1100 s'enviarà a la màquina remota *localhost* sobre el port 110, que és on escolta el servei POP3 (l'opció `-C` comprimeix el trànsit pel túnel).

Fer túnels sobre altres ports és molt fàcil. Per exemple, suposem que només tenim accés a un *remote proxy server* des d'una màquina remota (*remote login*), no des de la màquina local; en aquest cas, es pot fer un túnel per a connectar el navegador a la màquina local. Considerem que tenim *login* sobre una màquina passarel·la (*gateway*), que pot accedir a la màquina anomenada *proxy*, la qual executa l'*Squid Proxy Server* sobre el port 3128. Executem:

```
ssh -C -L 8080:proxy:3128 user@gateway
```

Després de connectar-nos, tindrem un túnel escoltant sobre el port local 8080 que reconduirà el trànsit des de *gateway* cap a *proxy* al 3128. Per a navegar de manera segura, només s'haurà de fer `http://localhost:8080/`.

1.3.3. VPN SSL (via *tun driver*)

OpenSSH v4.3 introdueix una nova característica que permet crear *on-the-fly* VPN via el túnel *driver* (*tun*) com un pont entre dues interfícies en diferents segments de xarxa a través d'Internet i per la qual cosa un ordinador (B en el nostre cas) “quedarà dins” de la xarxa de l'altre ordinador (A) malgrat estar dins d'una altra xarxa. Per a analitzar aquest mecanisme, considerarem que l'ordinador A i B estan connectats a la xarxa via Ethernet i a Internet a través d'una *gateway* que fa NAT. A té IP sobre la seva xarxa (privada) 10.0.0.100, i B sobre la seva xarxa (privada) 192.168.0.100 i, com vam dir, cadascun té accés a Internet NAT *gateway*.

A través d'OpenSSH, connectarem B a la xarxa d'A via una interfície túnel `ssh`. Com hem definit, A ja té IP sobre la xarxa A (10.0.0.100) i que es veu des del *gateway* extern com a adreça 1.2.3.4, és a dir aquesta és la IP pública del servidor `ssh` de A (l'encaminador haurà de fer un *forwarding* d'1.2.3.4:22 a 10.0.0.100:22 perquè externament es pugui contactar amb servidor `ssh` de A). B també ha de tenir una IP sobre la xarxa A que serà la seva interfície `tun0` i a la qual assignarem 10.0.0.200. B també ha de tenir accés al servidor `ssh` de A (o bé accés directe, o el port 22 ha de ser *forwarded* sobre la xarxa A en l'adreça 1.2.3.4). Quan el túnel estigui creat, B accedirà directament a la xarxa, la qual cosa significa que B “estarà” dins de la xarxa de A.

Els passos de configuració són els següents:

- 1) Activar l'IP *forwarding*: `echo 1 > /proc/sys/net/ipv4/ip_forward`
- 2) Túnel. Sobre B `ssh -w 0:0 1.2.3.4` es pot utilitzar com a opcions, a més, `-NTcf`. Això crea un *tunnel interface* `tun0` entre client (B) i el servidor (A), recordeu que A té adreça pública (1.2.3.4). S'haurà de tenir accés de *root* a tots dos sistemes perquè puguin crear les interfícies (en verificar A que es té en `sshd_config` `PermitRootLogin yes` i `PermitTunnel yes`). Es recomana utilitzar SSH *keys* (PKI), per la qual cosa s'haurà de canviar `PermitRootLogin` `without-password` i si sobre el client no es vol donar accés de *root*, es pot utilitzar l'ordre `sudo` (vegeu man `sudoers`).
- 3) Verificar les interfícies: `ip addr show tun0`
- 4) Configurar les interfícies:
Ordinador A: `ip link set tun0 up ip addr add 10.0.0.100/32 peer 10.0.0.200 dev tun0`
Ordinador B: `ip link set tun0 up ip addr add 10.0.0.200/32 peer 10.0.0.100 dev tun0`

- 5) Provar: sobre B `ping 10.0.0.100` i sobre A `ping 10.0.0.200`
- 6) *Routing* directe: tenim un enllaç que connecta B a la xarxa A tot i que és necessari inicialitzar el *routing*. Sobre B: `ip route add 10.0.0.0/24 via 10.0.0.200` (per a permetre enviar des de B a qualsevol màquina de la xarxa A).
- 7) *Routing invers*: també perquè els paquets tornin a B sobre A hem de fer `arp -sD 10.0.0.200 eth0`

El *routing* ens assegurarà que altres màquines que estan en la xarxa A puguin enviar paquets a B a través d'A. Podem provar fent des de B `ping 10.0.0.123`. Si es desitja que tots els paquets de B vagin a través de A, haurem de canviar el *default gateway* de B però això no és simple ja que el mateix túnel va per Internet. Primer hem de crear un *host-based route* cap a la màquina A (tots els paquets excepte els que creen el *link* han d'anar pel túnel, però òbviament no els que creen el túnel). Sobre B: `ip route add 1.2.3.4/32 via 192.168.0.1`. En aquest cas, 192.168.0.1 és el *default gateway* per a B, és a dir el *gateway* sobre la xarxa B que proveeix de connectivitat a Internet i que ens servirà per a mantenir el túnel. Després podem canviar el *default gateway* sobre B: `ip route replace default via 10.0.0.1`. Amb la qual cosa, tindrem que 192.168.0.1 és el *default gateway* de la xarxa B i 10.0.0.1 *default gateway* de la xarxa A. Ja que la màquina B està connectada a la xarxa d'A, estem indicant utilitzar la xarxa A com a *default gateway* en lloc de *default gateway* sobre la xarxa B (com seria habitual). Es pot verificar això amb l'ordre `tracert google.com`.

1.3.4. Túnel encadenats

Moltes vegades, l'accés a xarxes internes només és possible a través d'un servidor SSH (que estan en la DMZ) i des d'aquest és possible la connexió cap a servidors SSH interns per a després arribar a la màquina SSH del nostre interès (i si volem copiar arxius o fer el *forwarding* de X11, pot ser tot un repte). La manera de fer-ho és primer connectar-nos a l'M1, després d'aquesta a l'M2 (a la qual només és possible connectar-se des d'M1) i des d'aquesta a l'M3 (a la qual només és possible connectar-se des d'M2). Una manera de facilitar l'autenticació és utilitzar l'*agent forwarding* amb el paràmetre `-A` i posant la nostra clau pública en tots els `$HOME/.ssh/authorized_keys`. Així, quan anem executant les diferents ordres, la petició de claus pot ser *forwarded* cap a la màquina anterior i així successivament. Les ordres seran `ssh -A m1.org` i des d'aquesta `ssh -a m2.org` i al seu torn `ssh -a m3.org`. Per a millorar això, podem automatitzar-ho amb `ssh -A -t m1.org ssh -A -t m2.org ssh -A m2.org` (s'ha inclòs l'opció `-t` perquè `ssh` hi assigni una pseudo-tty i només veurem la pantalla final). L'*applet* SSHmenu pot ajudar a automatitzar tot això (<http://sshmenu.sourceforge.net/>).

Una forma de gestionar efectivament aquesta connexió "multisalts" és mitjançant l'opció `ProxyCommand` de `ssh` (vegeu `man ssh_config`), que ens

permetrà connectar-nos de manera més eficient. Per a això, definirem en *\$HOME/.ssh/config* les següents ordres:

```
Host m1
  HostName m1.org
Host m2
  ProxyCommand ssh -q m1 nc -q0 m2.org 22
Host m3
  ProxyCommand ssh -q m2 nc -q0 m3.org 22
```

On la primera ordre (quan fem `ssh m1`) ens connectarà a `m1.org`. Quan executem `ssh m2`, s'establirà una connexió `ssh` sobre `m1` però l'ordre `ProxyCommand` utilitzarà l'ordre `nc` per a estendre la connexió sobre `m2.org`. El tercer és una ampliació de l'anterior, per la qual cosa quan executem `ssh m3` és com si executéssim una connexió a `m1` i després ampliéssim aquesta a `m2` i després a `m3`.^[23]

1.4. Serveis de transferència de fitxers: FTP

L'FTP (*File Transfer Protocol*) és un protocol client/servidor (sota TCP) que permet la transferència d'arxius des d'un sistema remot i cap a un sistema remot. Un servidor FTP és un ordinador que executa el *daemon* `ftpd`.

Alguns llocs que permeten la connexió anònima sota l'usuari *anonymous* són generalment repositoris de programes. En un lloc privat, es necessitarà un usuari i una contrasenya per a accedir-hi. També és possible accedir a un servidor FTP mitjançant un navegador i generalment avui dia els repositoris de programes se substitueixen per servidors web (per exemple, Apache) o altres tecnologies com Bittorrent (que fa servir xarxes *peer to peer*, P2P) o servidors web amb mòduls de WebDav o l'ordre `scp` (*secure copy*) que forma part del paquet `openssh-client` o el parell `sftp/sftpd` que formen part dels paquets `openssh-client` i `openssh-server`, respectivament. No obstant això, es continua utilitzant en alguns casos i Debian, per exemple, permet l'accés amb usuari/contrasenya o la possibilitat de pujar arxius al servidor (si bé amb serveis `web-dav` també és possible fer-ho).

El protocol (i els servidors/clients que l'implementen) d'FTP per definició no és encriptat (les dades, usuaris i contrasenyes es transmeten en text net per la Xarxa), amb el risc que això representa. No obstant això, hi ha una sèrie de servidors/clients que suporten SSL i, per tant, encriptació.

1.4.1. Client FTP (convencional)

Un client FTP permet accedir a servidors FTP i hi ha una gran quantitat de clients disponibles. L'ús de l'FTP és summament simple; des de la línia d'ordres, executeu:


```
ftp nom-servidor
```

O també `ftp` i després, de manera interactiva: `open nom-servidor`

El servidor sol·licitarà un nom d'usuari i una contrasenya (si accepta usuaris anònims, s'introduirà *anonymous* com a usuari i la nostra adreça de correu electrònic com a contrasenya), i a partir del *prompt* de la línia d'ordres (després d'alguns missatges), podrem començar a transferir fitxers.

El protocol permet la transferència en mode ASCII o binari. És important decidir el tipus de fitxer que cal transferir perquè una transferència d'un binari en mode ASCII inutilitzarà el fitxer. Per a canviar d'un mode a un altre, s'han d'executar les ordres `ascii` o `binary`. Les ordres útils del client FTP són el `ls` (navegació en el directori remot), `get nom_del_fitxer` (per a descarregar fitxers) o `mget` (que admet `*`), `put nom_del_fitxer` (per a enviar fitxers al servidor) o `mput` (que admet `*`); en aquests dos últims, s'ha de tenir permís d'escriptura sobre el directori del servidor. Es poden executar ordres locals si abans de l'ordre s'insereix un `!`. Per exemple, `!cd /tmp` significarà que els arxius que baixin a la màquina local es descarregaran en `/tmp`. Per a poder veure l'estat i el funcionament de la transferència, el client pot imprimir marques, o *ticks*, que s'activen amb les ordres `hash` i `tick`. Hi ha altres ordres que es poden consultar en el full del manual (`man ftp`) o fent `help` dins del client. Tenim nombroses alternatives per als clients, per exemple en mode text: `ncftp`, `lukemftp`, `lftp`, `cftp`, `yafc` `Yafc` o, en manera gràfica: `gFTP`, `WXftp`, `LLNL XFTP`, `guiftp`.

Enllaç d'interès

En la Viquipèdia disposeu d'una comparativa de diversos clients FTP:
http://en.wikipedia.org/wiki/Comparison_of_FTP_client_programari

1.4.2. Servidors FTP

El servidor tradicional d'UNIX s'executa a través del port 21 i és engegat pel *daemon* `inetd` (o `xinetd`, segons es tingui instal·lat). En `inetd.conf` convé incloure el *wrapper* `tcpd` amb les regles d'accés en `host.allow` i el `host.deny` en la crida al `ftpd` per l'`inetd` per a incrementar la seguretat del sistema. Quan rep una connexió, verifica l'usuari i la contrasenya i el deixa entrar si l'autenticació és correcta. Un FTP *anonymous* treballa de manera diferent, ja que l'usuari només podrà accedir a un directori definit en l'arxiu de configuració i a l'arbre subjacent, però no cap amunt, per motius de seguretat. Aquest directori generalment conté directoris `pub/`, `bin/`, `etc/ilib/` perquè el *daemon* d'FTP pugui executar ordres externes per a peticions de `ls`. El *daemon* `ftpd` suporta els següents arxius per a la seva configuració:

- **/etc/ftpusers:** llista d'usuaris que no són acceptats en el sistema. Un usuari per línia.
- **/etc/ftphroot:** llista d'usuaris als quals se'ls substituirà el directori base `chroot` quan es connectin. Necessari quan desitgem configurar un servidor anònim.

Vegeu també

El tema de la seguretat del sistema s'estudia en el mòdul "Administració de seguretat".

- `/etc/ftpwelcome`: anunci de benvinguda.
- `/etc/motd`: notícies després del *login*.
- `/etc/nologin`: missatge que es mostra després de negar la connexió.
- `/var/log/ftpd`: *log* de les transferències.

Si en algun moment volem inhibir la connexió a l'FTP, es pot incloure l'arxiu `/etc/nologin`. L'`ftpd` mostra el seu contingut i acaba. Si existeix un arxiu `.message` en un directori, l'`ftpd` el mostrarà quan s'hi accedeixi. La connexió d'un usuari passa per cinc nivells diferents:

- 1) tenir una contrasenya vàlida;
- 2) no aparèixer en la llista de `/etc/ftpusers`;
- 3) tenir un *shell* estàndard vàlid;
- 4) si apareix en `/etc/ftpchroot`, es canviarà al directori `home` (també si és *anonymous* o FTP);
- 5) si l'usuari és *anonymous* o FTP, llavors haurà de tenir una entrada en el `/etc/passwd` amb usuari FTP, però podrà connectar-se especificant qualsevol contrasenya (per convenció, s'utilitza l'adreça de correu electrònic).

És important prestar atenció al fet que els usuaris habilitats únicament per a utilitzar el servei FTP no disposin d'un *shell* a l'entrada corresponent d'aquest usuari en `/etc/passwd` per a impedir que aquest usuari tingui connexió, per exemple, per `ssh` o `telnet`. Per a això, quan es creï l'usuari, caldrà indicar, per exemple:

```
useradd -d/home/nteum -s /bin/false nteum  
i després: passwd nteum,
```

la qual cosa indicarà que l'usuari `nteum` no tindrà *shell* per a una connexió interactiva (si l'usuari ja existeix, es pot editar el fitxer `/etc/passwd` i canviar l'últim camp per `/bin/false`). A continuació, s'ha d'agregar com a última línia `/bin/false` en `/etc/shells`. En [28] es descriu pas a pas com es crea tant un servidor FTP segur amb usuaris registrats com un servidor FTP *anonymous* per a usuaris no registrats. Dos dels servidors no estàndards més comuns són el ProFTPD i Vsftpd (tots dos inclosos en Debian, per exemple).

Per a fer la instal·lació del Proftpd sobre Debian, executeu `apt-get install proftpd`. Una vegada descarregat `debconf`, preguntarà si es vol executar per `inetd` o manualment (és recomanable triar l'última opció). Si s'ha d'aturar el servei (per a canviar la configuració, per exemple): `/etc/init.d/proftpd stop`. I per a modificar el fitxer: `/etc/proftpd.conf`. Un servidor (Debian)

Enllaços d'interès

Per a saber més sobre ProFTPD i Vsftpd, podeu visitar les següents pàgines:
<http://www.proftpd.org>
<https://security.appspot.com/vsftpd.html>

Enllaços d'interès

Per a configurar un servidor FTP en mode encriptat (TSL) o per a tenir accés *anonymous*, podeu consultar:
<http://www.debian-administration.org/articles/228>.
D'altra banda, per a saber més sobre la instal·lació i configuració de PureFtpd podeu consultar:
<http://www.debian-administration.org/articles/383>.

molt interessant és el PureFtpd (`pure-ftpd`), que és molt segur, permet usuaris virtuals, quotes, SSL/TSL i té un conjunt de característiques interessants. El paquet virtual de Debian `ftp-server` (<https://packages.debian.org/wheezy/ftp-server>) mostra tots els paquets d'aquest tipus inclosos en l'última versió.

1.5. *Active Directory Domain Controller amb Samba4*

Un dels aspectes més importants en la integració de sistemes és la gestió d'usuaris i identificació centralitzada, així com les autoritats d'autenticació i els permisos. En molt llocs basats en Windows, aquesta tasca és portada a terme per un *Active Directory* (AD, servei de directori en una xarxa distribuïda d'ordinadors (de vegades, també anomenat PDC per *Primary Domain Controller*), i és important disposar d'eines de la banda d'*Open Source* que permetin gestionar aquest tipus d'entorns. Samba versió 4 és una nova distribució d'aquest popular programari que permet la integració amb sistemes Windows actuant com a servidor d'arxius i/o impressores i a més, en aquesta última versió i de manera estable, pot actuar com a controlador d'un domini Windows acceptant clients, gestionant usuaris i directoris de manera centralitzada i totalment compatible.[29, 30]

Dels experts d'*Active Directory* (que generalment són consultors especialitzats i amb un alt cost), sabem que AD és una unió (que pot ser molt complicada d'entendre per als administradors que no estiguin dedicats al món Windows) de diferents serveis i tecnologies com ara DNS, Kerberos, LDAP i CIFS. Alguns hi inclouen DHCP també, però com veurem no és necessari en la nostra instal·lació. Estàriem en un error si penséssim que configurant cadascun d'aquests serveis tindríem el resultat del conjunt, però Samba4 presenta una capa d'abstracció, estable i eficient, que els integra per a oferir un producte complex però que es pot configurar i administrar seguint un conjunt de passos sense grans dificultats (encara que no s'ha de pensar que és simple). L'element crític (base dels majors problemes i errors) de l'AD és el DNS (en realitat, Windows utilitzarà l'AD com a DNS), ja que aquest el farà servir per a "agregar extraoficialment" a la llista de noms habituals en un DNS els servidors AD perquè els clients puguin trobar-los i tenir-hi accés.

En relació amb Kerberos i LDAP, els administradors de GNU/Linux saben de la seva potencialitat i complexitat, però en el cas d'AD estan integrats en el paquet i si bé els atorga una certa estandardització del sistema no són integrables amb servidors o altres clients, només s'utilitzen per als seus objectius i particularment quan fem servir Samba4, serà aquest qui els configurarà i gestionarà en el nostre nom amb petites modificacions per part de l'administrador. La versió actual de Samba (V4) no difereix de les anteriors quant a compartició d'arxius i impressores (fins i tot, s'ha simplificat la seva gestió/administració), i a més, amb la implementació d'AD permetrà que aquells administradors que utilitzin Windows puguin continuar utilitzant les seves eines de gestió i administració de domini només apuntant al servidor Samba.

Tota la configuració de Samba passarà ara per l'arxiu `smb.conf` (normalment en `/etc/samba/smb.conf`) amb definicions simplificades però permetent la gestió complexa d'un domini Windows mitjançant les eines (també complexes) de Windows com per exemple RSAT (*Remote Server Administration Tools*) i, òbviament, a través del sistema GNU/Linux i la CLI de Samba4 es tindrà accés a tota la configuració i administració de l'AD (sense necessitat d'utilitzar Windows en cap cas).[30, 33]

En la nostra instal·lació habitual, farem servir Debian Wheezy com a distribució, i si bé Samba4 està en els repositoris *backport* (en els de Wheezy només hi ha la versió 3.x però estarà disponible en la nova edició de Debian Jessie) optem, per la facilitat a l'hora de resoldre les dependències, per baixar el codi font i compilar-lo.

1.5.1. Passos preliminars

Una de les primeres verificacions que hem de fer és que NPT estigui instal·lat i funcionant correctament (`apt-get install ntp` i provar que se sincronitza correctament amb `ntpq -p`), ja que Kerberos és molt exigent quant als valors del rellotge. Un segon aspecte que cal cuidar és definir els servidors amb IP estàtiques (almenys durant la prova de concepte de la instal·lació i configuració de l'AD i, òbviament, en un servidor en producció). Per a això, definirem `/etc/network/interfaces` amb:

```
auto eth0
iface eth0 inet static
    address 192.168.1.33
    netmask 255.255.255.0
    gateway 192.168.1.1
```

Després, haurem de verificar el `/etc/hosts` (si és que no disposem d'un DNS on puguem modificar els registres A i PTR per a incloure el nostre servidor d'AD) per a agregar (en el nostre cas) una entrada com aquesta: `192.168.1.33 sysdw.nteum.org sysdw`. Com DNS, es podria fer servir qualsevol d'aquells que s'han vist anteriorment, però en aquest cas ens va semblar més adequat, per a fer una prova total de Samba4, utilitzar el que porta intern el paquet i que l'utilitzarà l'AD. Per aquest motiu, haurem d'estar segurs de desactivar qualsevol altre servei de DNS que tinguem en aquesta màquina perquè no hi hagi conflictes (mireu per exemple amb `netstat -anotup`). L'altre aspecte important és el nom de domini, i sense entrar en polèmiques en relació amb com es tracta en GNU/Linux i com es "veuen" en Windows, hem escollit per a simplificar el del DNS (si bé alguns experts indiquen que pot haver-hi conflictes quan tractem amb un DNS extern). És a dir, un *AD Domain* no significa realment un *DNS domain* sinó que és una "unió" híbrida de DNS i que Kerberos denomina *realm*. És per això que configurarem el `/etc/resolv.conf` apuntant com a DNS a la mateixa màquina de Samba4 i amb el domini DNS d'aquesta (i com a secundari un de públic). Recordeu que si es té instal·lat i actiu el

paquet `resolvconf` les modificacions de `/etc/resolv.conf` són dinàmiques i les inclusions dels DNS s'hauran de fer sobre `/etc/network/interfaces` amb els tags `dns-nameservers` i `dns-search`:

```
domain nteum.org
search nteum.org
nameserver 192.168.1.33
nameserver 8.8.8.8
```

A més a més, cal verificar que `/etc/hostname` coincideix amb el nom indicat en `/etc/hosts`. Finalment, com que Samba4 actuarà com a servidor d'arxius i com a PDC/AD, és necessari assegurar que el sistema d'arxius que estiguem exportant suporti ACL (*access control list*) i atributs estesos, i per a això modificarem `/etc/fstab` per a indicar al *file system* corresponent els atributs `user_xattr`, `acl`, incloent-hi una línia com:

```
UUID=.... / ext4 user_xattr,acl,errors=remount-ro 0 2.
```

Aquí ho hem fet sobre el directori *root* ja que tenim una partició només, però això només s'ha d'aplicar a aquelles particions que es desitgi servir als clients.

Finalment, reiniciarem la màquina perquè totes les configuracions s'actualitzin correctament.

1.5.2. Compilar Samba4

En primer lloc, hem de tenir una sèrie de paquets instal·lats (si bé molts d'aquests ja es troben instal·lats; aquesta llista pot variar segons les fonts que es consultin), però els requeriments de Samba4 indiquen:

```
apt-get install build-essential libacl1-dev libattr1-dev \
  libblkid-dev libgnutls-dev libreadline-dev python-dev libpam0g-dev \
  python-dnspython gdb pkg-config libpopt-dev libldap2-dev \
  dnsutils libbsd-dev attr krb5-user docbook-xsl libcups2-dev acl
```

Altres experts en Samba4 i AD també inclouen (algunes d'aquestes ja estaran incloses o es resoldran per dependències):

```
apt-get install pkg-config libacl1 libblkid1 attr libattr1 libgnutls-dev \
  libncurses-dev libdm0-dev libfam0 fam libfam-dev xsltproc libnss3-dev \
  docbook-xsl-doc-html docbook-xsl-ns python-dnspython libcups2-dev krb5-config
```

En el nostre cas, utilitzarem com a servidor de Kerberos el nostre servidor AD, per la qual cosa és important la seva configuració (paquets `krb5-user` i `krb5-config`) i haurèm d'introduir durant la seva instal·lació el domini (*realm*) que servirà en lletres majúscules, que en el nostre cas serà NTEUM.ORG, i el *password* haurà de tenir certa complexitat (com per exemple PaSSw0d2014) perquè el programa el consideri vàlid. Per a obtenir i compilar l'última versió de Samba4, farem:

wget <http://www.samba.org/samba/ftp/stable/samba-4.x.y.tar.gz> (s'ha de reemplaçar x.y amb l'última versió).

```
tar -xzf samba-4.x.y.tar.gz
cd samba-4.x.y
./configure --prefix=/opt/samba4
make
make install
```

Nosaltres hem escollit com a lloc d'instal·lació */opt/samba4*, per la qual cosa haurem d'actualitzar la variable PATH amb (s'ha de posar en *.profile*):

```
export PATH=/usr/local/samba/bin:/usr/local/samba/sbin:$PATH
```

i perquè la configuració quedi com és habitual, es crea un enllaç a */etc/samba* amb `ln -s /usr/local/samba/etc /etc/samba`. [32, 33]

El següent és aprovisionar l'*AD Domain*:

```
samba-tool domain provision --use-rfc2307 --interactive
```

Aquí haurem d'indicar el *realm* (NTEUM.ORG, que és el mateix que configurem en krb5) i el domini (NTEUM en el nostre cas). És important que el *password* coincideixi amb el que hem introduït en krb5 (i tingui certa complexitat), i assegurar-se que el *DNS forwarder* apunti el *DNS server* real que resoldrà les peticions de DNS i introduir la línia `allow dns updates = nonsecure` en els paràmetres globals. L'arxiu */etc/samba/smb.conf* quedarà com:

```
[global]
workgroup = NTEUM
realm = nteum.org
netbios name = SYSDW
server role = active directory domain controller
dns forwarder = 158.109.0.9
allow dns updates = nonsecure

[netlogon]
path = /opt/samba4/var/locks/sysvol/nteum.org/scripts
read only = No

[sysvol]
path = /opt/samba4/var/locks/sysvol
read only = No
```

A continuació, hem d'ajustar la configuració de Kerberos fent:

```
mv /etc/krb5.conf /etc/krb5.conf.org
cp /opt/samba4/private/krb5.conf .
```

El contingut de la qual haurà de quedar com:

```
[libdefaults]
default_realm = NTEUM.ORG
dns_lookup_realm = false
dns_lookup_kdc = true
```

Una vegada reiniciat el sistema, podem engegar el servidor simplement executant `samba` i mirar el `/var/log/syslog` per a verificar que s'ha arrencat correctament, i si hi ha errors, de quin tipus són per a solucionar-los (si executem `ps -edaf | grep samba` veurem aproximadament 14 processos samba si tot ha anat bé). Podrem verificar l'execució del servidor fent:

```
smbclient -L localhost -U%
  Domain=[NTEUM] OS=[Unix] Server=[Samba 4.1.9]
  Sharename      Type            Comment
  -----
  netlogon        Disk
  sysvol          Disk
  IPC$            IPC             IPC Service
  Domain=[NTEUM] OS=[Unix] Server=[Samba 4.1.9]
  Server          Comment
  -----
  Workgroup       Master
  -----

smbclient //localhost/netlogon -UAdministrator -c 'ls'      Ens sol·licitarà el passwd d'administrator
Enter Administrator's password:
Domain=[NTEUM] OS=[Unix] Server=[Samba 4.1.9]
.                  D              0   Wed Jul 23 10:11:33 2014
..                 D              0   Wed Jul 23 10:11:39 2014

44541 blocks of size 262144. 7486 blocks available
```

Per a verificar el DNS:

```
host -t SRV _ldap._tcp.nteum.org
  _ldap._tcp.nteum.org has SRV record 0 100 389 sysdw.nteum.org.

host -t SRV _kerberos._udp.nteum.org.
  _kerberos._udp.nteum.org has SRV record 0 100 88 sysdw.nteum.org.

host -t A sysdw.nteum.org
sysdw.nteum.org has address 158.109.65.67
```

També cal verificar que tenim connexió externa amb un `ping google.com`, per exemple.

Finalment, per a verificar el funcionament de KRB5 (el *realm* ha d'anar en majúscules):

```
kinit administrator@NTEUM.ORG
Password for administrator@NTEUM.ORG:
Warning: Your password will expire in 41 days on Wed Sep  3 17:10:52 2014
```

I podríem treure (si bé en entorns de producció no seria recomanable) la caducitat del *password* amb: `samba-tool user setexpiry administrator --noexpiry`

Finalment, si tot funciona haurem de repetir el procediment de recrear el domini però agregant `-use-ntvfs` com a paràmetre, per a això podem fer:

```
mv /etc/samba/smb.conf /etc/samba/smb.conf.org
samba-tool domain provision --realm nteum.org --domain NTEUM --adminpass PaSSw0d2014 \
    --server-role=dc --use-ntvfs
```

El nou arxiu *smb.conf* quedarà (agregant novament la línia `allow dns updates`):

```
[global]
workgroup = NTEUM
realm = nteum.org
netbios name = SYSDW
server role = active directory domain controller
dns forwarder = 158.109.0.9
server services = rpc, nbt, wrepl, ldap, cldap, kdc, drepl, winbind, ntp
    _signnd, kcc, dnsupdate, dns, smb
dcerpc endpoint servers = epmapper, wkssvc, rpcecho, samr, netlogon, lsa
rpc, spoolss, drsuapi, dssetup, unixinfo, browser, eventlog6, backupkey, dnsserv
er, winreg, srvsvc
allow dns updates = nonsecure
[netlogon]
path = /opt/samba4/var/locks/sysvol/nteum.org/scripts
read only = No

[sysvol]
path = /opt/samba4/var/locks/sysvol
read only = No
```

I que */etc/krb5.conf* és:

```
[libdefaults]
default_realm = NTEUM.ORG
dns_lookup_realm = false
dns_lookup_kdc = true
```

S'ha de reiniciar novament i executar `samba` (recordeu que encara no el tenim com a servei i que l'hem d'executar manualment). Si volem que Samba4 serveixi els directoris *homes*, haurem d'agregar en *smb.conf* una secció amb:

```
[home]
    path = /home/
    read only = no
```

Perquè l'administrador tingui els privilegis correctes en la gestió dels comptes des d'entorns Windows (i que Samba4 creï els directoris *home* automàticament), haurem d'executar:

```
net rpc rights grant 'NTEUM\Domain Admins' SeDiskOperatorPrivilege
-Uadministrator.
```

Ara hauríem de comprovar que podem configurar un client Windows i agregar-lo al directori. Per a això (en el nostre cas), utilitzarem una màquina Windows8 on en primer lloc modificarem la interfície de xarxa perquè el DNS faci referència al nostre servidor d'AD (anem a *Control Panel > Network and Internet*

> *Network and Sharing Center*, seleccionem el dispositiu de xarxa actiu i a propietats hi canviem el DNS, i deixem només l'IP del nostre AD). Després anem a *Control Panel* > *System and Security* > *System* i seleccionem *Advanced System Settings* i dins *Computer Name* > *Change* i introduïm *administrator* i *PaSSw0d2014* per a agregar la màquina al domini (en cas que no ens seleccioni el domini per defecte, podem posar a *Username*: *NTEUM\administrator*, i després el *passwd*), que ens donarà un missatge de confirmació si tot ha anat bé i ens demanarà fer un *reboot* de la màquina. Quan s'iniciï novament, podrem seleccionar un altre usuari/domini i connectar-nos a *NTEUM\administrator* (que és el compte prèviament definit). Amb tot això, es generaran una sèrie de passos en el client i el servidor per a aprovisionar en aquesta nova màquina la configuració i directori arrel.[32, 33]

Per a administrar el lloc AD, es poden utilitzar les eines de Windows (RSAT), les quals es poden obtenir (sense càrrec) des del lloc del fabricant i amb la guia indicada en [31] i exemples de configuració en [32]. També és possible utilitzar l'eina Swat (<https://wiki.samba.org/index.php/swat2>, última versió de novembre del 2012), però la seva instal·lació pot presentar alguns inconvenients (sobretot si Samba4 s'ha instal·lat des de les fonts). Finalment, en l'adreça <https://wiki.samba.org/index.php/samba4/InitScript> podrem trobar l'*script* per a iniciar i apagar el servidor AD de manera automàtica.

1.6. Serveis d'intercanvi d'informació en nivell d'usuari

1.6.1. El Mail Transport Agent (MTA)

Un MTA (*Mail Transport Agent*) s'encarrega d'enviar i rebre els correus des d'un servidor de correu electrònic cap a Internet i des d'Internet, que implementa el protocol SMTP (*Simple Mail Transfer Protocol*). Totes les distribucions incorporen diferents MTA i, per exemple, els de Debian es poden consultar en el seu paquet virtual *mail-transport-agent* (<https://packages.debian.org/wheezy/mail-transport-agent>). Una de les que s'utilitzen habitualment és *exim*, ja que és més fàcil de configurar que altres paquets MTA, com són *postfix* o *sendmail* (aquest últim és un dels precursors). *Exim* presenta característiques avançades com rebutjar connexions de llocs de *spam* coneguts, posseeix defenses contra correus brossa (*junk mails*) o bombardeig de correu (*mail bombing*), i és extremadament eficient en el processament de grans quantitats de correus.

La seva instal·lació és mitjançant `apt-get install exim4-daemon-heavy` (en aquest cas, s'optarà per la instal·lació de la versió *heavy*, que és la més completa i suporta llista d'accés, ACL, i característiques avançades, i en instal·lacions més senzilles es pot optar per *exim4-daemon-light*). La seva configuració es fa mitjançant `dpkg-reconfigure exim4-config`, on una resposta típica a les preguntes efectuades és:

- *General type of mail configuration: Internet site*; el correu és enviat i rebut utilitzat per SMTP.

- *System mail name*: remix.world (el nostre domini)
- *IP-addresses to listen on for incoming SMTP connections*: (deixar en blanc)
- *Other destinations for which mail is accepted*: remix.world
- *Domains to relay mail for*: (deixar en blanc)
- *Machines to relay mail for*: (deixar en blanc)
- *Keep number of DNS-queries minimal (Dial-on-Demand)?*: No
- *Delivery method for local mail*: Maildir format in home directory
- *Split configuration into small files?*: No

El servidor ja estarà configurat i es pot provar amb `echo test message | mail -s "test" adminp@SySDW.nteum.org` (canviant l'adreça, per descomptat) i verificar que el correu ha arribat a l'usuari adminp (els errors es poden trobar en `/var/log/exim4/mainlog`). La configuració serà emmagatzemada en `/etc/exim4/update-exim4.conf.conf`. Per a configurar autenticació per TLS, ACL i Spam Scanning, podeu consultar <https://wiki.debian.org/exim>.

1.6.2. External SMTP

Quan instal·lem un nou sistema com a servidors o estacions de treball, un aspecte rellevant és el servidor de correu i podem instal·lar grans paquets com els ja esmentats Postfix, Exim o Zimbra (en la seva versió Community <http://www.zimbra.com/>), fent que els correus cap a dominis externs utilitzin els serveis externs d'SMTP (per exemple, els de Google). Per a màquines virtuals, estacions de treball o portàtils és una mica més complicat ja que generalment tenen IP privades o en xarxes internes, per la qual cosa és necessari tenir un servidor que faci de receptor dels correus externs al nostre domini, és a dir, un servidor que faci les funcions de *smarthost*, per exemple el Google Apps SMTP. Per a detalls de la seva configuració, es pot seguir la documentació d'<https://wiki.debian.org/gmailandexim4>. D'acord amb la informació de Google (<https://support.google.com/a/answer/2956491?hl=es>) i per a comptes gratuïts, el nombre màxim de destinataris permès per domini i dia és de 100 missatges i Gmail reescriurà l'adreça del remitent. Per a la seva configuració, executarem `dpkg-reconfigure exim4-config` i seleccionarem:

- *mail sent by smarthost; received via SMTP or fetchmail*.
- *System mail name*: localhost
- *IP-addresses to listen on for incoming SMTP connections*: 127.0.0.1
- *Other destinations for which mail is accepted*: (deixar en blanc)
- *Machines to relay mail for*: (deixar en blanc)
- *IP address or host name of the outgoing smarthost*: smtp.gmail.com::587
- *Hide local mail name in outgoing mail?*: No
- *Keep number of DNS-queries minimal (Dial-on-Demand)?*: No
- *Delivery method for local mail*: mbox format in /var/mail
- *Split configuration into small files?*: Yes

Aquesta és la configuració més adequada si no es té un IP visible externament. L'enviament sobre el port 587 de Gmail utilitza STARTTLS per a assegurar la protecció del passwd i per a indicar l'usuari i passwd d'accés a Gmail (utilitzeu un compte només per a aquest objectiu, no el compte habitual de Gmail), s'ha d'editar el fitxer `/etc/exim4/passwd.client` i agregar la següent línia.

```
*.google.com:SMTPAccountName@gmail.com:yOuRpaSsw0RD
```

Després, executar (per a evitar que altres usuaris de la màquina puguin llegir el seu contingut):

```
chown root:Debian-exim /etc/exim4/passwd.client
chmod 640 /etc/exim4/passwd.client
```

Gmail reescriurà l'adreça del remitent automàticament, però si no ho fes, o enviem un *smarthost* que no ho fa, hauríem de configurar l'arxiu `/etc/email-addresses` amb totes les combinacions d'adreces possibles per a utilitzar (una per línia) i l'adreça que es reescriurà (per exemple, `nteum@remix.world: Smt-paccountname@gmail.com`). Després, s'haurà d'executar:

```
update-exim4.conf
invoke-rc.d exim4 restart
exim4 -qff
```

Amb això s'actualitza i recarrega la configuració, i es força a enviar tots els correus que estan pendents. Com vam mostrar amb anterioritat, en el fitxer `/var/log/exim4/mainlog` tindrem els errors si n'hi ha. Si existeixen errors d'autenticació sobre Gmail, hem de verificar amb `host smtp.gmail.com` quins són els *hosts* que retorna i si aquests concorden amb el patró inclòs en `/etc/exim4/passwd.client`. Si és diferent, canvieu-lo perquè coincideixi.

1.7. Internet Message Access Protocol (IMAP)

Aquest servei suportat pel *daemon* `imapd` (els actuals suporten el protocol IMAP4rev1) permet accedir a un arxiu de correu electrònic (*mail file*) que es troba en una màquina remota. El servei `imapd` es presta mitjançant els ports 143 (`imap2`), 220 (`imap3`) o 993 (`imaps`) quan suporta encriptació per SSL. Si s'utilitza `inetd`, aquest servidor s'enrega mitjançant una línia en el fitxer `/etc/inetd.conf` com:

```
imap2 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
imap3 stream tcp nowait root /usr/sbin/tcpd /usr/sbin/imapd
```

En aquest exemple, es crida el *wrapper* `tcpd` que funciona amb `hosts.allow` i `hosts.deny` per a incrementar la seguretat. Les aplicacions més populars són `courier-imap`, `cyrus-imapd`, `dovecot-imapd`, entre d'altres. Per a provar que el servidor `imap` funciona, es podria utilitzar un client, per exemple Thunderbird/Icedove (Debian), Evolution, Squirrelmail, o qualsevol altre client que suporti IMAP, crear un compte per a un usuari local, configurar-lo

de manera adequada perquè es connecti sobre la màquina local i verificar el funcionament de `imap`.

Com a prova de concepte, podem fer la instal·lació de `apt-get install dovecot-imapd` que amb les opcions per defecte permet una connexió encriptada per SSL i sobre bústies *mailbox* (o si volem, sobre bústies *maildir* hauríem de canviar la configuració de `/etc/dovecot/conf.d/10-mail.conf`). Dovecot és un servidor molt potent, per la qual cosa permet una gran quantitat d'opcions i configuracions (consulteu <http://wiki2.dovecot.org/>) i un resum aplicat d'aquestes en Debian Wheezy*. Les proves es poden completar configurant Evolution perquè es connecti al nostre servidor/usuari i llegir els correus del servidor prèviament configurat. És important notar que alguns clients de correu/servidors d'Imap només suporten el format *mailBox* i no *maildir*, i per aquest motiu s'ha de tenir en compte quan s'utilitzin els clients/servidors d'Imap. En les activitats que hem fet fins ara, tant `exim4` com `dovecot-imapd` suporten tots dos formats i s'han de configurar durant la instal·lació.

*<https://workaround.org/ispmail/wheezy/setting-up-dovecot>

1.7.1. Aspectes complementaris

Suposem que com a usuaris, tenim quatre comptes de correu en servidors diferents i volem que tots els missatges que arriben a aquests comptes es recullin en un només, al qual puguem accedir externament, i que hi hagi també un filtre de correu brossa (*antispam*). Primer s'han d'instal·lar `exim4 + imapd` i comprovar que funcionen.

Per a recollir els missatges de diferents comptes, s'utilitzarà `Fetchmail`, (que s'instal·la amb `apt-get install fetchmail`). A continuació, s'ha de crear el fitxer `.fetchmailrc` en el nostre `$HOME` (també es pot utilitzar l'eina `fetchmailconf`), que haurà de ser una cosa semblant a:

```
set postmaster "adminp"
set bouncemail
set no spambounce
set flush
poll pop.domain.com proto pop3
  user 'nteum' there with password 'MyPaSSwOrD' is 'nteum' here
poll mail.domain2.com
  user 'adminp' there with password 'MyPaSSwOrD' is 'adminp' here
  user 'nteum' there with password 'MyPaSSwOrD' is 'nteum' here
```

L'acció `set` indica a `Fetchmail` que aquesta línia conté una opció global (enviament d'errors, eliminació dels missatges dels servidors, etc.). A continuació, s'especifiquen dos servidors de correu: un perquè comprovi si hi ha correu amb el protocol POP3 i un altre perquè provi a fer servir diversos protocols amb la finalitat de trobar-ne un que funcioni. Es comprova el correu de dos usuaris amb la segona opció de servidor, però tot el correu que es trobi s'envia a l'*spool* de correu de `adminp`. Això permet comprovar diverses bústies de diferents servidors, com si es tractés d'una única bústia. La informació específica de cada usuari comença amb l'acció `user`. El `Fetchmail` es pot posar en el

cron (per exemple, en `/var/spool/cron/crontabs/fetchmail` agregant
`1 * * * * /usr/bin/fetchmail -s`) perquè s'executi automàticament o
 executar-lo en mode *daemon* (poseu `set daemon 60` en `.fetchmailrc` i exe-
 cuteu-lo una vegada, per exemple, en autostart de Gnome/KDE o en el
`.bashrc` i s'executarà cada 60 segons).

Per a treure el correu brossa, es farà servir SpamAssassin.

Aquesta configuració s'executarà mitjançant Procmail, que és una eina molt
 potent en la gestió del correu (permet repartir el correu, filtrar-lo, reenviar-lo
 de manera automàtica, etc.). Un cop instal·lat (`apt-get install procmail`),
 s'ha de crear un fitxer anomenat `.procmailrc` en el home de cada usuari, que
 cridarà l'SpamAssassin:

Podeu instal·lar SpamAssassin
 mitjançant `apt-get install
 spamassassin.`

```
# Poseu yes per a missatges de funcionament o depuració
VERBOSE=no
# Considerem que els missatges estan en "~/Maildir", canviar si és un altre
PATH=/usr/bin:/bin:/usr/local/bin:
MAILDIR=$HOME/Maildir
DEFAULT=$MAILDIR/

# Directori per a emmagatzemar els fitxers
PMDIR=$HOME/.procmail
# Comentar si no volem log de Procmail
LOGFILE=$PMDIR/log
# filtre de Smap
INCLUDEDRC=$PMDIR/spam.rc
```

L'arxiu `~/procmail/spam.rc` conté:

```
# si l'SpamAssassin no és en el PATH, agregueu a la variable PATH el directori
:Ofw: spamassassin.lock
| spamassassin -a

# Les tres línies següents mouran el correu spam a un directori anomenat
# "spam-folder". Si es vol guardar el correu en la safata d'entrada,
# per a després filtrar-lo amb el client, comenteu les tres línies.

:O:
* ^X-Spam-Status: Yes
spam-folder
```

L'arxiu `~/spamassassin/user_prefs` conté algunes configuracions útils
 per a SpamAssassin (consulteu la bibliografia).

```
# user preferences file. Vegeu man Mail::SpamAssassin::Conf

# Llindar per a reconèixer un Spam.
# Default 5, però amb 4 funciona una mica millor
required_hits 4

# Llocs dels quals considerem que mai arribarà Spam
whitelist_from root@debian.org
whitelist_from *@uoc.edu

# Llocs dels quals sempre arriba SPAM (separats per comes)
blacklist_from viagra@dominio.com
```

```
# les direccions en Whitelist i Blacklist són patrons globals com:
# "amigo@lugar.com", "*@isp.net", o "*.domain.com".

# Inserteu la paraula SPAM en el subject (facilita fer filtres).
# Si no es desitja comentar la línia.
subject_tag [SPAM]
```

Això generarà un *tag* `X-Spam-Status: Yes` en la capçalera del missatge si es creu que el missatge és *spam*. Després, s'haurà de filtrar i posar a una altra carpeta o esborrar-lo directament. Es pot fer servir el `procmail` per a filtrar missatges de dominis, usuaris, etc. Finalment, es pot instal·lar un client de correu i configurar els filtres perquè seleccionin tots els correus amb `X-Spam-Status: Yes` i els esborri o els envii a un directori. Després, verificarem els falsos positius (correus identificats com a brosa però que no ho són). Un aspecte complementari d'aquesta instal·lació és que si es desitja tenir un servidor de correu a través de correu web (*webmail*), és a dir, poder consultar els correus del servidor mitjançant un navegador sense haver d'instal·lar un client ni configurar-lo (com consultar un compte de Gmail o Hotmail), és possible instal·lar Squirrelmail (`apt-get install squirrelmail`) per a donar aquest servei.

Enllaç d'interès

Hi ha altres possibilitats com instal·lar MailDrop en lloc de Procmail, Postfix en lloc d'Exim, o incloure Clamav/Amavisd com a antivirus (Amavisd permet vincular Postfix amb SpamAssassin i Clamav). Per a saber més sobre aquest tema, podeu visitar la següent pàgina web: <http://www.debian-administration.org/articles/364>.

Enllaç d'interès

Per a més informació sobre `procmail` i el filtrat de missatges, consulteu: <http://www.debian-administration.org/articles/242>.

Enllaç d'interès

Sobre Squirrelmail en Debian, consulteu: <http://www.debian-administration.org/articles/200>.

1.8. Grups de discussió

Les *news* o grups de discussió són suportats mitjançant el protocol NNTP. Instal·lar un servidor de grups de discussió és necessari quan es desitja llegir *news* fora de línia, quan es vol tenir un repetidor dels servidors centrals o es vol un propi servidor mestre de *news*. Els servidors més comuns són INN o CNEWS, però són paquets complexos i destinats a grans servidors. Leafnode és un paquet USENET que implementa el servidor TNP, especialment indicat per a llocs amb grups reduïts d'usuaris, però on es desitja accedir a gran quantitat de grups de notícies. Aquest servidor s'instal·la en la configuració bàsica de Debian i es poden reconfigurar amb `dpkg-reconfigure leafnode` paràmetres com ara els servidors centrals, el tipus de connexió, etc. Aquest *daemon* s'engega des de `inetd` de manera similar al `imap` (o amb `xinetd`). Leafnode suporta filtres mitjançant expressions regulars indicades (del tipus `^Newsgroups:.*[.]alt.flame$`) en `/etc/news/leafnode/filters`, on per a cada missatge es compara la capçalera amb l'expressió regular i, si hi ha coincidència, el missatge es rebutja.

La configuració d'aquest servidor és simple i tots els arxius han de ser propietat d'un usuari de *news* amb permís d'escriptura (s'ha de verificar que aquest propietari existeix en `/etc/passwd`). Tots els arxius de control, *news* i la configuració es troben en `/var/spool/news`, excepte la configuració del propi servidor que està en el fitxer `/etc/news/leafnode/config`. En la configu-

ració, hi ha alguns paràmetres obligatoris que s'han de configurar (per exemple, perquè el servidor pugui connectar-se amb els servidors mestres), com `server` (servidor de *news* des d'on s'obtiniran i enviaran les *news*) i `expire` (nombre de dies als quals un fil o sessió s'esborrarà després d'haver estat llegit). Tenim, així mateix, un conjunt de paràmetres opcionals d'àmbit general o específic del servidor que podrien configurar-se. Per a més informació, consulteu la documentació (`man leafnode` o `/usr/doc/leafnode/README.Debian`). Per a verificar el funcionament del servidor, es pot fer `telnet localhost nntp` i, si tot funciona correctament, sortirà la identificació del servidor i es quedarà esperant una ordre. Com a prova, es pot introduir `help` (per a avortar, feu `Ctrl+[` i després `Quit`).

1.9. World Wide Web (httpd)

Apache és un dels servidors més populars i amb majors prestacions d'HTTP (*HyperText Transfer Protocol*). Apache té un disseny modular i suporta extensions dinàmiques de mòduls durant la seva execució. És molt configurable quant al nombre de servidors i de mòduls disponibles i suporta diversos mecanismes d'autenticació, control d'accés, *metafiles*, *proxy caching*, servidors virtuals, etc. Amb mòduls (inclosos en Debian) és possible tenir PHP3, Perl, Java Servlets, SSL i altres extensions*.

*Podeu consultar la documentació en <http://www.apache.org>.

Apache està dissenyat per a executar-se com un procés *daemon standalone*. En aquesta forma, crea un conjunt de processos fills que gestionaran les peticions d'entrada. També pot executar-se com *Internet daemon* mitjançant `inetd`, per la qual cosa s'engegarà cada vegada que es rebí una petició, però no és recomanat. La configuració del servidor pot ser extremadament complexa segons les necessitats (consulteu la documentació); no obstant això, aquí veurem una configuració mínima acceptable. La seva instal·lació és simple, per exemple en Debian, `apt-get install apache2 apache2-doc apache2-utils`. La configuració del servidor estarà en `/etc/apache2` i per defecte el *RootDirectory* en `/var/www`. Després de la seva instal·lació, s'engegarà i posant com URL en un navegador `http://localhost` veurem que funciona (ens mostrarà el famós **It works!**). Hi ha 5 ordres que hauran d'estar a la ment de tot administrador:

- `a2enmod`|`a2dismod` per a habilitar/deshabilitar mòduls,
- `a2ensite`|`a2dissite` per a habilitar/deshabilitar llocs (virtuals) i
- `apachectl` per a gestionar la configuració del servidor (`start`|`stop`|`restart`|`graceful`|`graceful-stop`|`configtest`|`status`|`fullstatus`|`help`).

La configuració d'Apache2 en Debian és una mica diferent de la distribució general ja que intenta facilitar al màxim la configuració del servidor quant a mòduls, *hosts* virtuals i directives de configuració (no obstant això, ràpidament es poden trobar les equivalències amb altres distribucions). Els principals arxius que es troben en el directori `/etc/apache2/` són `apache2.conf`, `ports.conf` i cinc directoris `mods-available`|`mods-enabled`, `sites-available`|`sites-enabled` i `conf.d`.

Per a informació addicional, podeu llegir `/usr/share/doc/apache2.2*` i en particular `/usr/share/doc/apache2.2-common/README.Debian`.

- 1) *apache2.conf* és l'arxiu principal de configuració on es defineixen en un nivell funcional les prestacions del servidor i es criden els arxius de configuració corresponents (*ports*, *conf.d*, *sites-enabled*). Es recomana posar com a sufix *.load* per als mòduls que hagin de ser carregats i *.conf* per a les configuracions, però hi ha regles més extenses quant als sufixos/noms que poden ampliar-se en la documentació (p. ex., s'ignoren tots els arxius que no comencen per lletra o nom).
- 2) *ports.conf* (s'inclou en l'arxiu de configuració global) defineix els ports on s'atendran les connexions entrants, i quins d'aquests són utilitzats en els *hosts* virtuals.
- 3) Els arxius de configuració en *mods-enabled/* i *sites-enabled/* són per als llocs actius i els mòduls que desitgen ser carregats en el servidor. Aquestes configuracions s'activen creant un enllaç simbòlic des dels directoris respectius **-available/* fent servir `a2enmod/a2dismod`, `a2ensite/a2dissite`.
- 4) Els arxius de *conf.d* són per a configuracions d'altres paquets o agregats per l'administrador i es recomana que acabin amb *.conf*.
- 5) Perquè sigui efectiva la configuració per defecte en aquests directoris, *apache2* ha de ser gestionat a través de `/etc/init.d/apache2` o `service` o `apache2ctl`.
- 6) L'arxiu *envvars* és el que contindrà la definició de les variables d'entorn i és necessari modificar bàsicament dos USER/GROUP que seran amb les quals s'executarà i obtindrà els permisos. Per defecte es crea l'usuari *www-data* i el grup *www-data* (es poden canviar). Per aquest motiu, s'haurà de fer servir `APACHE_RUN_USER=www-data` i `APACHE_RUN_GROUP=www-data`.

Apache també pot necessitar integrar diversos mòduls en funció de la tecnologia que suporti, per la qual cosa s'hauran d'agregar les biblioteques/paquets corresponents, per exemple:

- 1) Perl: `apt-get install libapache2-mod-perl2`
- 2) Rugby: `apt-get install libapache2-mod-ruby`
- 3) Python: `apt-get install libapache2-mod-python`
- 4) MySQL in Python: `apt-get install python-mysqldb`
- 5) PHP: `apt-get install libapache2-mod-php5 php5 php-pear php5-xcache`
- 6) PHP with MySQL: `apt-get install php5-mysql`

1.9.1. Servidors virtuals

Per *servidors virtuals* s'entenen els llocs aliats que seran servits cadascun de manera independent de l'altre amb els seus propis arxius i configuració. En primer lloc, deshabilitarem el lloc per defecte amb `a2dissite default`. Els llocs que crearem seran `remix.world` i `lucix.world`, que disposaran de dos arxius de configuració en `/etc/apache2/sites-available/` anomenats com el domini.

Contingut de l'arxiu `/etc/apache2/sites-available/remix.world.conf`

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    ServerName remix.world
    ServerAlias www.remix.world
    DocumentRoot /var/www/remix/
    ErrorLog /var/log/apache2/remix-error.log
    CustomLog /var/log/apache2/remix-access.log combined
    Options ExecCGI # habilitar Script en Perl
    AddHandler cgi-script .pl
</VirtualHost>
```

Contingut de l'arxiu `/etc/apache2/sites-available/lucix.world.conf`

```
<VirtualHost *:80>
    ServerAdmin adminpSySDW.nteum.org
    ServerName lucix.world
    ServerAlias www.lucix.world
    DocumentRoot /var/www/lucix/
    ErrorLog /var/log/apache2/lucix-error.log
    CustomLog /var/log/apache2/lucix-access.log combined
    Options ExecCGI # habilitar Script en Perl
    AddHandler cgi-script .pl
</VirtualHost>
```

Aquesta configuració és molt bàsica i l'estudiant haurà de consultar la informació detallada en [14]. Com es pot observar, els directoris arrel per a cada domini estaran en `/var/www/remix|lucix` i els arxius de *log* en `/errors/accessos` en `/var/log/apache2/mmmmm-error.log` i `var/log/apache2/nnnnn-access.log/`. Per a crear els directoris `mkdir -p /var/www/remix;mkdir -p /var/www/lucix` i en els quals es podria posar un `index.html` amb alguna identificació que mostres quin domini s'està carregant. Per exemple, per a `remix.world`:

```
<html><body><h1>REMIX: It works!</h1>
<p>This is the default web page for this server.</p>
<p>The web server software is running but no content has been added, yet.</p>
</body></html>
```

I el mateix per a `lucix.world`, però canviant la línia en `<h1></h1>`. Per als *logs* no hem de fer res ja que el directori `/var/log/apache2` ja existeix i els arxius els crearà el servidor. Finalment, hem d'activar els llocs (crear l'enllaç des de *sites-available* a *sites-enabled*) amb `a2ensite remix.world.conf; a2ensite lucix.world.conf` i reiniciar `apache2` amb `service apache2 reload`. Com que no disposem dels dominis en un DNS primari, podem editar `/etc/hosts` i agregar per a la IP del nostre servidor (p. ex., 192.168.1.37) dues línies:

```
192.168.1.37 remix.world
192.168.1.37 lucix.world
```

Després, des d'un navegador podrem introduir l'URL `remix.world` i el resultat serà la visualització de `l'index.html` que ens dirà: **REMIX: It works!**

Un dels avantatges d'Apache és que pot agregar funcionalitat mitjançant mòduls especialitzats i que es trobaran en `/etc/apache2/mods-available/`. Per a obtenir la llista de mòduls disponibles per a Apache podem fer, per exemple, `apt-cache search libapache2*`, i per a instal·lar-lo `apt-get install [module-name]`, els quals estaran disponibles per al seu ús (recordeu que pot ser necessària alguna configuració addicional en els arxius del lloc). Podem mirar els disponibles amb `ls -al /etc/apache2/mods-available/` i instal·lar-lo amb `a2enmod [module-name]`. Per a posar en una llista els mòduls carregats, podem fer servir `/usr/sbin/apache2 -M` que ens posarà en una llista amb *shared* els carregats dinàmicament i amb *static* els que es troben compilats amb el servidor (aquests es poden obtenir també amb `apache2 -l`). Els mòduls en el directori *mods-available* tenen extensions *.load* (indica la biblioteca que cal carregar) i *.conf* (configuració addicional del mòdul), però quan utilitzem l'ordre `a2enmod` només s'ha d'indicar el nom del mòdul sense extensió. Per a deshabilitar un mòdul, `a2dismod [module-name]`.

Com a mostra d'aquestes propietats, configurarem un lloc segur (https) sota el domini `remix.world` però que redirigirem al directori `/var/www/remix.ssl`. En primer lloc, crearem un certificat (autosignat) per al nostre lloc amb

```
make-ssl-cert /usr/share/ssl-cert/ssleay.cnf /etc/ssl/private/remix.crt
```

indicant-hi el domini que volem validar (`remix.world` en el nostre cas) –només cal introduir el domini i deixar els àlies en blanc– i si no podem executar `make-ssl-cert`, hem d'assegurar-nos que tenim el paquet `ssl-cert`. Després activem el mòdul SSL amb `a2enmod ssl`, creem el directori `/var/www/remix.ssl` i modifiquem `l'index.html` com vam fer amb els anteriors. Després, creem la configuració del lloc (podem utilitzar la que ve per defecte i modificar-la):

```
cd /etc/apache2/sites-available; cp default-ssl remix.world.ssl.conf
```

Editem l'arxiu `remix.world.ssl.conf` (només mostrem les línies principals/modificades):

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerAdmin admin@sydw.nteu.org
    DocumentRoot /var/www/remix.ssl
    <Directory />
        Options FollowSymLinks
        AllowOverride None
    </Directory>
    <Directory /var/www/remix.ssl>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
    </Directory>
# línies igual que l'arxiu original...
```

```
ErrorLog $APACHE_LOG_DIR/remix.world.ssl_error.log
CustomLog $APACHE_LOG_DIR/remix.world.ssl_access.log combined

SSLEngine on
SSLCertificateFile /etc/ssl/private/remix.crt
#SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
# línies igual que l'arxiu original...
</VirtualHost>
</IfModule>
```

Finalment, ens queda activar el lloc (`a2ensite remix.world.ssl.conf`), reiniciar Apache2 (amb `service apache2 reload`) i des del navegador fer `https://remix.world` que, com que el certificat és autosignat, ens farà un advertiment i acceptarem el certificat i haurem d'obtenir **SSL - REMIX: It works!**

Un aspecte interessant és la funció de l'arxiu `.htaccess`* en els directoris del nostre domini. Aquest arxiu es pot utilitzar per al control d'accés al lloc (p. ex., habilitar/restringir IP), control d'accés a carpetes, llistes, reencaminaments (p. ex., a una altra pàgina/site, a una altra carpeta, a un altre domini, a https, etc.), evitar el *hotlinking* (per a evitar que ens facin enllaços a fitxers, generalment vídeos, que consumeixen amplada de banda del nostre servidor), canviar la pàgina per defecte, crear URL amigables, afavorir el cau del nostre lloc, etc. Com a mostra d'això, per a evitar per exemple que una carpeta sigui inaccessible, n'hi ha prou amb posar un arxiu `.htaccess` en la mateixa amb el contingut `deny from all`. Per a permetre que una carpeta del nostre lloc (p. ex., del domini `remix.com/valid-user`) tingui accés amb un usuari i `passwd`, haurem de crear dins d'aquesta un arxiu `.htaccess` amb el següent contingut (també podem crear un `index.html` modificat dins d'aquesta carpeta per a verificar-ne el funcionament):

*<http://httpd.apache.org/docs/2.2/howto/htaccess.htm>

```
AuthName "Restricted Area"
AuthType Basic
AuthUserFile /etc/apache2/htpasswd
AuthGroupFile /dev/null
require valid-user
```

Per a crear l'usuari, fem `htpasswd -c /etc/apache2/htpasswd adminp` que ens demanarà el `passwd` per a aquest usuari i l'emmagatzemarà en l'arxiu indicat. Després, posem com a URL `http://remix.world/valid-user/`. Ens demanarà l'usuari (`adminp`) i el `passwd` que emmagatzemem i veurem **REMIX->Valid-User: It works!** En cas contrari, ens continuarà demanant l'usuari `/passwd` i si fem *Cancel* no indicarà un missatge d'*Authorization Required* impeding l'accés.

1.9.2. Apache + PHP + Mysql + PhpMyAdmin

Una qüestió important per als servidors web dinàmics és aprofitar els avantatges d'Apache PHP (un llenguatge de programació usat generalment per a la creació de contingut per a llocs web) i una base de dades com MySQL incloent

un programa administrador de MySQL com PHPMyAdmin, tot això funcionant conjuntament. Les distribucions han evolucionat molt i en Debian és summament fàcil engegar aquest conjunt (però tampoc representa cap dificultat descarregar-se el programari font, compilar-lo i instal·lar-lo si es desitja, per exemple, tenir les últimes versions dels paquets per algun motiu però recordeu que implicarà més treball i dedicació).

En primer lloc, instal·lem PHP amb `apt-get install php5`, que ens indicarà que canviarà la manera de treballar d'Apache2 instal·lant-ne un altre. Això es produeix perquè la configuració per defecte d'Apache treballa amb una eina anomenada *MPM-worker*. Aquest és un mòdul de multiprocessament que pot manejar múltiples peticions ràpidament utilitzant múltiples *threads* per procés client. No obstant això, aquest no és compatible amb algunes extensions PHP i per això l'*MPM-worker* és reemplaçat per *MPM-prefork*, que permet manejar totes les peticions PHP (en mode compatibilitat) i evitar que si una petició falla pugui afectar altres peticions. Hi ha un altre mòdul anomenat *mpm-itk* (<http://mpm-itk.sesse.net/>) que és similar a *prefork* però té millors prestacions i gestió de permisos (consulteu la bibliografia en apache.org). Per a verificar que PHP funciona, creem un fitxer per exemple dins de *RootDirectory* de *remix.world* anomenat *test.php* amb el següent contingut: `<?php phpinfo() ?>`, i si en l'URL introduïm `http://remix.world/test.php` haurem de veure una taula amb la versió i informació sobre el paquet PHP instal·lat.

Per instal·lar els paquets MySQL i PHPMyAdmin, farem `apt-get install mysql-server` (és molt important que recordeu la contrasenya d'accés que introduïm, però sempre podem fer `dpkg-reconfigure mysql-server`; tenint en compte que perdrem tot el que hi hagi en la BD). També hi ha altres mètodes (menys agressius) per a recuperar la contrasenya del *root*). Després, per a instal·lar PHPMyAdmin farem `apt-get install phpmyadmin` i prestar atenció, ja que ens demanarà la clau d'accés per a entrar en la base de dades i crear una clau d'accés per a entrar en l'aplicació via navegador. Després, podrem posar en l'URL del nostre navegador `http://localhost/phpmyadmin`, ens sol·licitarà l'usuari (*root* generalment) i el *passwd* que hem introduït i ja podrem gestionar el servidor de bases de dades MySQL.

1.9.3. Altres servidors httpd

Lighttpd és un servidor web (amb llicència BSD) dissenyat per a ser ràpid, segur, flexible, que implementa la majoria d'estàndards i està optimitzat per a entorns on la velocitat és molt important (consumeix menys CPU/RAM que altres servidors) i és molt apropiat per a qualsevol servidor que hagi de donar suport a grans càrregues. Entre les seves principals característiques, hi ha la de *virtual hosting*, reencaminaments http i reescriptures d'URL, donar suport a CGI, SCGI i FastCGI, PHP, Ruby, Python entre d'altres i a més amb consum de memòria constant.

La seva instal·lació en Debian és `apt-get install lighttpd`, i si tenim Apache sobre el port 80 ens donarà un error. Per a això, haurem d'editar l'arxiu `/etc/lighttpd/lighttpd.conf` i canviar la línia `server.port = 8080` i reiniciar `service lighttpd start`. Des del navegador, es pot escriure l'adreça `http://localhost:8080index.lighttpd.html` i llavors veurem la pàgina inicial de lighttpd. Per defecte, Lighttpd té el seu directori arrel en `/var/www` (en Debian) i l'arxiu de configuració en `/etc/lighttpd/lighttpd.conf`. Configuracions addicionals són en `/etc/lighttpd/conf-available` i poden ser habilitades amb l'ordre `lighttpd-enable-mod`, la qual crea enllaços entre `conf-enabled` i `conf-available`. Es poden deshabilitar amb `lighttpd-disable-mod`.

Per a habilitar el servidor de FastCGI per a executar PHP, haurem d'instal·lar PHP-FPM amb la instrucció `apt-get install php5-fpm php5` i sobre l'arxiu `/etc/php5/fpm/php.ini` treure el comentari a la línia `cgi.fix_pathinfo=1`. Després haurem d'activar el servidor PHP-FPM, per la qual cosa farem una còpia de l'arxiu original i el modificarem:

```
cd /etc/lighttpd/conf-available/  
cp 15-fastcgi-php.conf 15-fastcgi-php-spawnfcgi.conf
```

Modificar `15-fastcgi-php.conf` amb:

```
# -*- depends: fastcgi -*-  
  
# Start an FastCGI server for php  
fastcgi.server += ( ".php" =>  
    (  
        "socket" => "/var/run/php5-fpm.sock",  
        "broken-scriptfilename" => "enable"  
    )  
)
```

Per a habilitar fastcgi, haurem de carregar els mòduls `lighttpd-enable-mod fastcgi` i `lighttpd-enable-mod fastcgi-php`, la qual cosa ens crea els enllaços corresponents, que amb la instrucció `ls` podem visualitzar: `ls -l /etc/lighttpd/conf-enabled`. Després, podem reiniciar amb la instrucció `service lighttpd force-reload`. Per a visualitzar si el servidor i FastCGI funcionen, creem un arxiu `/var/www/info.php` amb el següent contingut `<?php phpinfo(); ?>` i podrem visualitzar la pàgina de configuració de PHP on indica com Server API = FPM/FastCGI (`http://localhost:8080/info.php`).

Un altre servidor molt utilitzat actualment és **Nginx** (`http://nginx.org/`) programat en C i llicència BSD. Les seves funcions principals són com a servidor web/*proxy* invers de molt alt rendiment (pot suportar més de 10.000 connexions simultànies) i també pot funcionar com a *proxy* per a protocols de correu electrònic (IMAP/POP3). És un servidor utilitzat per grans instal·lacions

(WordPress, Netflix, Hulu, GitHub i parts de Facebook, entre d'altres) i entre les seves principals característiques hi ha (a més de servidor d'arxius estàtics, índexs i autoindexat i *proxy* invers amb opcions de cau) el balanç de càrrega, tolerància a fallades, SSL, FastCGI, servidors virtuals, *streaming* d'arxius (FLV i MP4.8) i suport per a autenticació, compatible amb IPv6 i SPDY. La seva instal·lació bàsica és simple, i per a la seva configuració, es pot consultar la wiki de `nginx` en <http://wiki.nginx.org/configuration>.

1.10. Servidor de WebDAV

El nom WebDAV són les sigles de *Web Based Distributed Authoring and Versioning* (també es refereix al grup de treball d'*Internet Engineering Task Force*) i és un protocol que permet que el web es transformi en un mitjà llegible i editable i proporciona funcionalitats per a crear, canviar i moure documents en un servidor remot (típicament, un servidor web). Això s'utilitza sobretot per a permetre l'edició dels documents que envia un servidor web, però també es pot aplicar a sistemes d'emmagatzematge generals basats en el web i als quals es pot accedir des de qualsevol lloc. En aquest subapartat, instal·larem un servidor WebDAV sobre Apache. El procés és el següent:

- 1) Verificar que tenim instal·lat Apache2 i, si no, fer la seva instal·lació com hem vist anteriorment i verificar que funciona (`apt-get install apache2`).
- 2) Habilitar els mòduls d'Apache que són necessaris per a WebDAV: `a2enmod dav_fs` i `a2enmod dav`.
- 3) Crear el directori per al directori virtual (podem fer, per exemple, `mkdir -p /var/www/webdav`) i permetre que Apache sigui el propietari del directori `chown www-data /var/www/webdav/`
- 4) Crear l'arxiu `/etc/apache2/sites-available/webdav.conf` per a definir la funcionalitat del servidor (en aquesta configuració, estem configurant tot el servidor com a WebDAV però podria ser un servidor virtual i en mode SSL per a major seguretat):

```
<VirtualHost *:80>
  ServerAdmin admin@sySDW.nteum.org
  DocumentRoot /var/www/webdav/
  ErrorLog /var/log/apache2/webdav-error.log
  CustomLog /var/log/apache2/webdav-access.log combined
  <Directory /var/www/webdav>
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    allow from all
    DAV On
    AuthName "Restricted WeDav Area"
    AuthType Basic
    AuthUserFile /etc/apache2/htpasswd
    AuthGroupFile /dev/null
    require valid-user
  </Directory>
</VirtualHost>
```

Enllaç d'interès

Sobre la integració de WebDAV amb Apache, podeu consultar l'article "WebDAV on Apache2" disponible en: <http://www.debian-administration.org/articles/285>

Es pot comprovar que la configuració és correcta amb

```
apache2ctl configtest.
```

5) Com s'ha fet anteriorment, es creen els usuaris indicant el `-c` si és el primer usuari que cal crear (`htpasswd [-c] /etc/apache2/htpasswd usuari`).

6) Es reinicia Apache perquè llegeixi la configuració `/etc/init.d/apache2 reload` i ja ens podem connectar a `http://localhost`, després de fer l'autenticació.

7) Des d'un GNU/Linux, podem provar la funcionalitat del servidor obrint el `nautilus` (gestor de fitxers) i des del menú *File->Connect to Server* podem seleccionar Servidor WebDAV introduint les dades (IP, directori, usuari, passwd) i ja tindrem accés com si d'una carpeta local es tractés.

8) Des de MacOS, podem utilitzar el mateix procediment que l'anterior des del gestor d'arxius o instal·lar un client específic (igualment per a Windows). El més recomanat per a això és CyberDuck (<http://cyberduck.io/>), que té llicència GPL i és una excel·lent aplicació (suporta múltiples protocols) i molt fàcil de configurar.

9) Una altra forma de provar-ho és amb un client WebDAV (en mode text), per exemple Cadaver*, amb `apt-get install cadaver`. A continuació, ens connectem al servidor amb `cadaver IP-nom del servidor`, i després d'autenticar-nos, podem crear un directori (`mkdir`), editar un arxiu, fer la llista d'un directori (`ls`), canviar de directori (`cd`), canviar els permisos d'execució (`chexec`), esborrar-lo (`rm`), etc.

*<http://www.webdav.org/cadaver>

En moltes ocasions, i atès que estarem fent transferències d'arxius, és important preservar la privadesa, per la qual cosa seria adequat treballar amb WebDAV però sobre SSL. La seva configuració no implica majors complicacions i veurem una manera diferent de generar els certificats (seran autosignats, fins que puguem tenir la nostra pròpia entitat certificadora) per a un domini en particular, en el nostre cas `webdav.world`. Per a això fem:

1) Ens canviem al directori on emmagatzemarem els certificats:

```
cd /etc/ssl/private
```

Fem la petició del certificat:

```
openssl req -config /etc/ssl/openssl.cnf -new -out webdav.csr
```

Aquesta ordre ens demanarà un *passwd* i una sèrie d'informació que quedarà en el certificat, però la més important és *Common Name* (CN), que serà on validarà el certificat (en el nostre cas, `webdav.world`).

Podem verificar la petició amb:

```
openssl req -in /etc/ssl/private/webdav.csr -noout -text
```

2) Creem la clau (*key*):

```
openssl rsa -in privkey.pem -out webdav.key
```

3) Signem:

```
openssl x509 -in webdav.csr -out webdav.crt -req -signkey webdav.key  
-days 3650
```

Ho podem verificar amb:

```
openssl x509 -noout -in /etc/ssl/private/webdav.crt -text
```

4) Generem un certificat en format DER: `openssl x509 -in webdav.crt -out webdav.der.crt -outform DER` Es pot verificar amb: `openssl x509 -in /etc/ssl/private/webdav.der.crt -inform der -noout -text`

5) Ara, generem l'arxiu de configuració d'Apache a partir de l'anterior: `cd /etc/apache2/sites-available; cp webdav.conf webdav-ssl.conf`
Modifiquem per a incloure les següents quatre línies a l'inici i modificar el VirtualHost:

```
<VirtualHost *:443>  
ServerName webdav.world  
SSLEngine on  
SSLCertificateFile /etc/ssl/private/webdav.crt  
SSLCertificateKeyFile /etc/ssl/private/webdav.key
```

...

Només ens queda activar el lloc (`a2ensite webdav-ssl.conf`), reiniciar Apache (`service apache2 restart`) i verificar que funciona en l'adreça `https://webdav.world/` prèvia acceptació del certificat (recordar posar una entrada en `/etc/hosts` amb el nom del domini i l'IP de la màquina similar a com es va fer en `remix.world`).

1.11. Servei de *proxy*: Squid

Un servidor *proxy* (*proxy server*, PS) s'utilitza per a estalviar amplada de banda de la connexió de xarxa, millorar la seguretat i incrementar la velocitat per a obtenir pàgines de la xarxa (*web-surfing*).

Squid és un *proxy caching server* per a web i dona suport als protocols HTTP, HTTPS, FTP, entre d'altres. Aquest redueix l'amplada de banda, millora el temps de resposta emmagatzemat en cau i reutilitza les pàgines més freqüents. Squid té un extens conjunt de regles de control que permeten optimitzar el flux de dades entre el client i el servidor aportant seguretat i control i encaminant les peticions correctament, la qual cosa permet el seu control i millora la utilització de l'amplada de banda de la xarxa. Squid té diferents modes de funcionament, però com a més importants podem esmentar *Forward Proxy* (és la manera bàsica sobre la qual es configura tota la resta), *Transparent* o *Interception Proxy* (permet incloure un *proxy* en una xarxa sense que els clients hagin de configurar res) i *Reverse Proxy* o *Accelerator-mode* (permet executar Squid per a millorar la resposta d'una granja de servidors web).

Per a instal·lar Squid com a *proxy-cache* (<http://www.squid-cache.org/>), en Debian fem `apt-get install squid3` i editarem l'arxiu de configuració `/etc/squid3/squid.conf` per a portar a terme una configuració bàsica. S'ha de tenir en compte que Squid és molt potent, però això es tradueix en una configuració que pot ser complexa; com a idea, simplement hem de considerar que l'arxiu de configuració, que està molt ben explicat, té aproximadament 5.700 línies (no obstant això, si executem

```
grep -v "^#" /etc/squid3/squid.conf | awk '$1 != "" {print $0}'
```

podrem veure que la configuració bàsica són unes 40 línies mentre que la resta són comentaris i opcions comentades).[16][17]

Definir l'ACL (*access control list*) per a habilitar la xarxa/IP que desitgem fer de *proxy*, en la línia 718 agreguem: `acl lan src 192.168.1.0/24`

Permetre l'accés, en la línia 842 (aprox.) agreguem: `http_access allow lan`
Canviar el port d'accés (línia 1136), per defecte és `http_port 3128` i es pot deixar que sigui l'estàndard per a Squid o posar el que es prefereixi (per exemple, 8080).

Agregar les regles de control, en la línia 3449 (aprox.):

```
request_header_access Referer deny all
request_header_access X-Forwarded-For deny all
request_header_access Via deny all
request_header_access Cache-Control deny all
```

Definim el nom visible, línia 3748 (aprox.): `visible_hostname remix.world`

Modifiquem la visibilitat de l'IP, línia 5541 (aprox.): `forwarded_for off`

Finalment, reiniciem el servidor: `service squid3 restart`

Ens donarà un missatge similar a `[ok] Restarting Squid HTTP Proxy 3.x: squid3[....] Waiting.....done.` (trigarà uns segons).

Amb el servidor en marxa, podem configurar els navegadors perquè utilitzin el *proxy*. Per exemple, en IceWeasel/Firefox en l'apartat de *Preferences/Options->Advanced->Network->Proxy* podem introduir les dades del nostre servidor. Sobre Chrome, per exemple en Windows, s'ha d'anar a *Settings->Advanced->Change proxy setting->Connections->Lan Settings* i introduir les dades del nostre servidor.

Una altra de les opcions que permet Squid és actuar com a *reverse proxy*, que és un tipus de *proxy* on les dades es recuperen d'un servidor i retornen als clients com si s'originessin en el *proxy*, de manera que els servidors queden ocults als clients com es mostra en la figura*. Això permet fer polítiques de balanç de

*http://upload.wikimedia.org/wikipedia/commons/6/67/reverse_proxy_h2g2bob.svg

càrrega i que les peticions estiguin en un únic domini, malgrat que internament poden estar distribuïdes en diversos servidors. Per a la seva configuració, hem de fer:

Especificar l'adreça del servidor intern, en la línia 1136 modificar:

```
http_port 80 defaultsite=192.168.1.33
```

Agregar `cache_peer`, en la línia 1937 (aprox.) agregar:

```
cache_peer 192.168.1.33 parent 80 0 no-query originserver
```

Canviar l'acl per a permetre qualsevol connexió, en la línia 842 (aprox.) modificar `http_access allow all`

Finalment, reiniciem el servidor: `service squid3 restart`

Quan ens connectem a l'IP/domini del servidor *proxy*, en realitat veurem les pàgines enviades pel servidor 192.168.1.33. Com a prova de concepte (si volem fer la prova amb una única màquina), podem posar, en lloc del servidor 192.168.1.33, un servidor extern (p. ex., `debian.org` o la seva IP), i quan posem com a URL el del nostre domini, visualitzarem la pàgina de `debian.org`.

Altres de les configuracions àmpliament utilitzades és *interception proxy* (o *transparent proxy*), que intercepta la comunicació normal a la capa de xarxa sense necessitat de configuracions específiques en el client, i per la qual cosa no sabran que estan darrere d'un *proxy*. Generalment, un *transparent proxy* està normalment localitzat entre el client i Internet amb el *proxy* fent les funcions de *router* o *gateway*. És habitual en les institucions que desitgen filtrar algun trànsit dels seus usuaris, ISP per a fer cau i estalviar amplada de banda o països que controlen l'accés a determinats llocs per part dels seus ciutadans. Per a implementar-lo sobre una institució, l'habitual és disposar d'una màquina amb dues interfícies de xarxa amb una de connectada a la xarxa interna i una altra cap a la xarxa externa. Els passos per a la seva configuració estan descrits en <http://wiki.squid-cache.org/configexamples/intercept/linuxdnat>:

Sobre `squid.conf`:

Modificar la `acl/http_access` sobre les IP, IP/Mask permeses com en el primer cas.

Configurar el port/mode com `http_port 3129 transparent` o en Squid 3.1+ s'haurà d'utilitzar `http_port 3129 intercept` per a interceptar paquets DNAT.

Sobre `/etc/sysctl.conf` modificar:

Permetre el *packet forwarding*: `net.ipv4.ip_forward = 1`

Controlar el *source route verification*: `net.ipv4.conf.default.rp_filter = 0`

No acceptar *source routing*: `net.ipv4.conf.default.accept_source_route = 0`

Considerant que l'IP de *proxy* està en la variable SQUIDIP i el port està en SQUIDPORT, incloure les següents regles de DNAT:

```
iptables -t nat -A PREROUTING -s $SQUIDIP -p tcp -dport 80 -j ACCEPT
iptables -t nat -A PREROUTING -p tcp -dport 80 -j DNAT -to-destination
    $SQUIDIP:$SQUIDPORT
iptables -t nat -A POSTROUTING -j MASQUERADE
iptables -t mangle -A PREROUTING -p tcp -dport $SQUIDPORT -j DROP
```

1.11.1. *Proxy* SOCKS

SOCKS (abreujament de SOCKetS) és un protocol d'Internet (en el model OSI estaria en una capa intermèdia entre la d'aplicació i la de transport) que permet a les aplicacions en mode client-servidor travessar de manera transparent un *firewall* de xarxa. Els clients que hi ha darrere d'un *firewall*, els quals necessiten accedir als servidors de l'exterior, poden connectar-se en el seu lloc a un servidor *proxy* SOCKS. Aquest servidor *proxy* controla quin client pot accedir al servidor extern i passa la petició al servidor. SOCKS pot ser utilitzat també de la forma contrària, la qual cosa permet que els clients de fora del *firewall* (clients externs) es connectin als servidors de dins del *firewall* (servidors interns).[26][27]

S'ha de tenir en compte que SOCKS només serveix en mode client/servidor, per la qual cosa un usuari ha de tenir instal·lat un client SOCKS, ja sigui en l'aplicació (com Firefox, Chrome) o dins de la pila TCP/IP des d'on el programari del client redirigeix els paquets en un túnel SOCKS. El procediment habitual comença quan el client SOCKS (p. ex., intern en una xarxa privada) inicia una connexió a un servidor SOCKS (el protocol SOCKS permet l'autenticació i el registre de les sol·licituds de connexió) i aquest (servidor SOCKS) actuar com a client IP per a la sol·licitud de connexió (del client intern en nom seu), la qual cosa significa que el servidor extern només serà conscient de les peticions del servidor SOCKS (així que actuarà com a *proxy forwarding*).

La pregunta habitual és si SOCKS resol de manera diferent el problema d'accés extern mitjançant NAT. La resposta és sí, ho fa de manera diferent, ja que els paquets en NAT només modifiquen les adreces (p. ex., canvien les IP privades per les pública del *router* com succeeix en un *router* ADSL) i el servidor rep/contesta les peticions. La sessió IP s'estableix directament des del client al servidor i el *router*/FW només modifica/filtra el paquet però no hi ha autenticació ni inspecció del paquet/aplicació/dades (es podria fer, però és complex). Els avantatges de SOCKS rau en el fet que proveeix autenticació per a protocols que no ho permeten, pot traspasar el *routing* per defecte d'una xarxa interna. Si bé HTTP i Telnet suporten autenticació per *firewall* (p. ex., utilitzant *Authenticated Proxy* on a Cisco *firewall*), els protocols encriptats mai poden ser autenticats per un FW, i en canvi SOCKS si pot fer-ho. No obstant això, hi ha

desavantatges, ja que el client ha de tenir interfície a SOCKS, el SO client ha de tenir interfície a SOCKS (per a interceptar el trànsit i reexpedir-lo al SOCKS *proxy*) i necessitarem un servidor SOCKS específic per a aquesta fi.

Un cas d'ús habitual és si considerem que estem en un punt de connexió sense fil oberta (p. ex., Wi-Fi) i no es desitja enviar dades de navegació sobre text net o es vol accedir a una pàgina web filtrada per *router*/FW perimetral. Una solució molt simple és utilitzar SSH (que inclou un servidor SOCKS) que pot xifrar tot el trànsit de navegació pel web i reencaminar-lo a través d'un equip de confiança quan s'està en algun altre punt de la xarxa. Per a això, haurem de disposar d'un servidor SSH perquè actuï com a representant en un ordinador remot (que li permeti connectar-se al mateix a través de SSH) i un client d'SSH en l'equip que està utilitzant. El que es farà amb un *proxy* és la creació d'un *middle-person* entre l'usuari i Internet. El navegador farà les sol·licituds de pàgines web al servidor *proxy*, que controla la sol·licitud i obté la pàgina des d'Internet i les retorna al client. El lloc web en realitat pensa que la sol·licitud prové del servidor *proxy*, no de l'equip que l'ha originat ocultant l'adreça IP d'origen. A més, la connexió entre l'ordinador i el *proxy* que passa a través d'SSH és xifrada i això evita que algú pugui obtenir els paquets des de la Wi-Fi (*sniffers* de Wi-Fi) en el lloc de la connexió.

Per a la seva configuració des d'on ens connectem, hem de tenir accés a un servidor SSH, sobre el qual crearem un túnel que passarà el trànsit web entre la nostra màquina local i el *proxy* SOCKS sobre SSH. Per a això, executem sobre la nostra màquina `ssh -ND 9999 login@remote-server.org` on haurem de reemplaçar `login@remote-server.org` amb l'usuari i nom o IP del servidor remot. El que està fent aquesta ordre és un *port forwarding* a través del port 9999 (pot ser qualsevol altre, però convé que sigui superior a 1024 per a evitar que només ho pugui fer *root*) i la connexió es reenvia a través d'un canal segur on el protocol d'aplicació s'utilitza per a determinar on connectar des de la màquina remota. Actualment, OpenSSH suporta els protocols SOCKS4-5 i per això actuarà com un servidor SOCKS. A continuació, se sol·licitarà el *passwd* i una vegada autenticat no passarà res (el -N indica que no obri un *prompt* interactiu, però continuarà funcionant). Si pel *firewall* només podem sortir pel port 443, per exemple, hauríem de configurar el *ssh server* per a escoltar pel port 443 i en el seu lloc executar `ssh -ND 9999 login@remote-server.org -p 443`. Ara és necessari configurar el client per a connectar-se al *proxy*, per exemple Firefox: *Options* -> *Advanced* -> *Network* -> *Connection* i seleccionar SOCKS, com a nom del servidor *localhost* (o el seu nom real si en té) i el port (9999) i guardar els ajustos i verificar que podem navegar sense problemes. Es pot utilitzar el *plugin* Foxy-Proxy* per a Firefox, que permet canviar entre el *proxy* i la connexió directa en funció del lloc o d'un control. Com a mesura addicional (d'anonimat), es pot configurar el servidor *proxy* per a resoldre peticions DNS en lloc del mètode habitual en Firefox posant com a URL `about:config` i modificant `network.proxy.socks_remote_dns=true`. També per a connexions lentes es pot utilitzar l'opció -C de *ssh* per a fer servir la compressió de SSH per *gzip*. En Thunderbird o altres clients, la configuració és similar.

*<https://addons.mozilla.org/es/firefox/addon/foxyproxy-standard>

Si el túnel deixa de funcionar (acostuma a ocórrer en xarxes molt ocupades), es pot utilitzar el paquet `autossh` en lloc del `ssh` per a establir la connexió que s'encarregarà de mantenir el túnel obert reiniciant automàticament la connexió. Un altre paquet interessant és `tsocks` (<http://tsocks.sourceforge.net/>), que es pot utilitzar quan el client que desitgem utilitzar no suporta el protocol SOCKS. `tsocks` monitora la trucada d'inici de sessió d'una aplicació (*connect*) i redirecciona la comunicació cap al *server* SOCKS sense que l'aplicació tingui cap informació. Per a això, s'ha d'instal·lar `tsocks` i configurar el *proxy* SOCKS que haurà d'utilitzar en el fitxer `/etc/tsocks.conf` indicant-hi els valors (p. ex., `server = 127.0.0.1`, `server_type = 5`, `server_port = 9999`). Després, bastarà a cridar l'aplicació amb `tsocks` aplicació o simplement l'ordre que obrirà una nova *shell* redirigida al *proxy* i per la qual cosa tot el que s'executi allà serà enviat al *proxy* SOCKS.

1.12. OpenLdap (LDAP)

LDAP significa *Lightweight Directory Access Protocol* i és un protocol per a accedir a dades basades en un servei X.500. Aquest s'executa sobre TCP/IP i el directori és similar a una base de dades que conté informació basada en atributs. El sistema permet organitzar aquesta informació de manera segura i utilitza rèpliques per a mantenir la seva disponibilitat, la qual cosa assegura la coherència i la verificació de les dades accedides-modificades.

El servei es basa en el model client-servidor, on existeixen un o més servidors que contenen les dades; quan un client es connecta i sol·licita informació, el servidor respon amb les dades o amb un punter a un altre servidor d'on podrà extreure més informació; no obstant això, el client només veurà un directori d'informació global [28, 19]. Per a importar i exportar informació entre servidors `ldap` o per a descriure una sèrie de canvis que s'aplicaran al directori, el format utilitzat es diu LDIF (*LDAP Data Interchange Format*). LDIF emmagatzema la informació en jerarquies orientades a objectes que després seran transformades al format intern de la base de dades. Un arxiu LDIF té un format similar a:

```
dn: o = UOC, c = SP o: UOC
objectclass: organization
dn: cn = Remix Nteum, o = UOC, c = SP
cn: Remix Nteum
sn: Nteum
mail: nteumuoc.edu
objectclass: person
```

Cada entrada s'identifica amb un nom indicat com a DN (*Distinguished Name*). El DN consisteix en el nom de l'entrada més una sèrie de noms que el relacionen amb la jerarquia del directori i on existeix una classe d'objectes (*objectclass*) que defineix els atributs que es poden fer servir en aquesta entrada. LDAP proveeix d'un conjunt bàsic de classes d'objectes: **grups** (inclou llistes desordenades d'objectes individuals o grups d'objectes), **localitzacions**

(com països i la seva descripció), **organitzacions i persones**. Una entrada pot, a més, pertànyer a més d'una classe d'objecte, per exemple, un individu és definit per la classe persona, però també pot ser definit per atributs de les classes `inetOrgPerson`, `groupOfNames` i `organization`.

L'estructura d'objectes del servidor (anomenat *schema*) determina quins són els atributs permesos per a un objecte d'una classe (que es defineixen en el fitxer `/etc/ldap/schema` com `inetorgperson.schema`, `nis.schema`, `opeldap.schema`, `corba.schema`, etc.). Totes les dades es representen com un parell atribut = valor, on l'atribut és descriptiu de la informació que conté; per exemple, l'atribut utilitzat per a emmagatzemar el nom d'una persona és `commonName`, o `cn`, és a dir, una persona anomenada Remix Nteum es representarà per `cn: Remix Nteum` i portarà associat altres atributs de la classe persona com `givenname: Remix` `surname: Nteum` `mail: nteum@uoc.edu`. A les classes hi ha atributs obligatoris i optatius i cada atribut té una sintaxi associada que indica quin tipus d'informació conté l'atribut, per exemple, `bin` (*binary*), `ces` (*case exact string*, ha de buscar-se igual), `cis` (*case ignore string*, poden ignorar-se majúscules i minúscules durant la cerca), `tel` (*telephone number string*, s'ignoren espais i '-') i `dn` (*distinguished name*). Un exemple d'un arxiu en format LDIF podria ser:

```
dn: dc = UOC, dc = com
objectclass: top
objectclass: organizationalUnit
```

```
dn: ou = groups, dc = UOC, dc = com
objectclass: top
objectclass: organizationalUnit
ou: groups
```

```
dn: ou = people, dc = UOC, dc = com
objectclass: top
objectclass: organizationalUnit
ou: people
```

```
dn: cn = Remix Nteum, ou = people, dc = UOC, dc = com
cn: Remix Nteum
sn: Nteum
objectclass: top
objectclass: person
objectclass: posixAccount
objectclass: shadowAccount
uid:remix userpassword:{crypt}p1pss2ii(0pgbs*do& = )eksd uidnumber:104
gidnumber:100
gecos:Remix Nteum
loginShell:/bin/bash
homeDirectory: /home/remix
shadowLastChange:12898
shadowMin: 0
shadowMax: 999999
shadowWarning: 7
shadowInactive: -1
shadowExpire: -1
shadowFlag: 0
```

```
dn:
```

```

cn = unixgroup, ou = groups, dc = UOC, dc = com
objectclass: top
objectclass: posixGroup
cn: unixgroup
gidnumber: 200
memberuid: remix
memberuid: altre-usuari

```

Les línies llargues es poden continuar a sota començant per un espai o un tabulador (format LDIF). En aquest cas, s'ha definit la base DN per a la institució `dc = UOC`, `dc = com`, la qual conté dues subunitats: `people` i `groups`. A continuació, s'ha descrit un usuari que pertany a `people` i a `group`. Una vegada preparat l'arxiu amb les dades, aquest ha de ser importat al servidor perquè estigui disponible per als clients LDAP. Hi ha eines per a transferir dades de diferents bases de dades a format LDIF [19]. Sobre Debian, s'ha d'instal·lar el paquet `slapd` i `ldap-utils`, que és el servidor d'OpenLdap i un conjunt d'utilitats per a accedir a servidors locals i remots. Durant la instal·lació, sol·licitarà un *passwd* per a gestionar l'administració del servei, es generarà una configuració inicial i s'engegarà el servei que es pot verificar amb l'ordre `slapcat` que, prenent la informació disponible (FQDN de `/etc/hosts`), generarà una cosa similar a aquesta:

```

dn: dc=nteum,dc=org
objectClass: top
objectClass: dcObject
objectClass: organization
o: nteum.org
dc: nteum
...
dn: cn=admin,dc=nteum,dc=org
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator   userPassword:: e1NTSEF9TU9jVE1qWlIPVFBmd2FiZWJtSjcrY0pYd2wvaTk5aUc=
...

```

Amb les ordres `ldapadd` i `ldapdelete`, es poden agregar/esborrar registres de la taula del servei utilitzant normalment arxius de text (sobretot per l'`add`) on estiguin definits els nous registres. Si bé no és un procediment complex per a gestionar el servidor, pot ser més adequat, en els primers passos, utilitzar algunes de les aplicacions que permeten gestionar el servidor d'una manera més amigable com per exemple `phpldapadmin` (administració per mitjà d'`apache+php`), `jxplorer` (aplicació en java), `gosa` o `lat` (tots aquests en la majoria de distribucions). En el nostre cas instal·larem `phpldapadmin`, que combina simplicitat i permet gestionar la majoria d'opcions del servidor Ldap.

Per a instal·lar-lo, fem `apt-get install phpldapadmin`, el qual ja reiniciarà el servidor Apache però, si no, fa que es reiniciï `/etc/init.d/apache2 restart`. Per a connectar-nos a un navegador, introduïm la direcció web: `http://localhost/phpldapadmin/`. Sortirà la pàgina de benvinguda i en la banda esquerra podrem fer el *login* (si el `ServerName` d'Apache no coincideix amb el LDAP, no demanarà com a usuari DN, per la qual cosa li haurèm d'indicar el correcte `cn=admin`, `dc=nteum`, `dc=org` i la contrasenya del servi-

dor LDAP). Una vegada connectat al servidor, seleccionem l'arrel (`dc=nteum, dc=org`) i seleccionem entre els *templates* que apareixen una *Organizational Unit (OU)* que anomenarem `users`, repetim els passos (opció `create new entry` des de l'arrel) i creem una altra *OU* que anomenem `groups`. Ara només ens queda crear els usuaris i els grups i assignar els usuaris als seus grups. Dins de la unitat organitzativa `groups`, crearem els grups `vendes (gid=1001)` i `compres (gid=1002)` i en la unitat organitzativa `users` crearem els usuaris `juan pirulo (uid=1001, vendes, id=jpirulo)` i `ana pirulo (uid=1002, compres, id=apirulo)`. Per als grups, seleccionem l'OU `groups`, fem `create new child` i seleccionem `Posix Group`. Crearem els dos grups indicats però haurem de modificar el `gid` ja que els assigna per defecte a partir de 1.000 i nosaltres els volem diferents. Repetim l'operació en `users` seleccionant `User Account`. Aquí ens demanarà les dades dels usuaris (nom, cognom, grup, contrasenyes, *home directory*, *shell*, etc.) i després haurem de canviar el `uid`, ja que els assigna per defecte. Després, podem observar novament el contingut executant `slapcat` en les noves dades generades. `Phpldapadmin` es pot configurar amb l'arxiu `/usr/share/phpldapadmin/config/config.php`, però en la majoria de casos la configuració per defecte ja és operativa (en alguns casos, pot ser necessari adaptar la línia `$servers->setValue('server','base',array('dc=nteum,dc=org'))`; per a adequar a les dades del nostre servidor).

Per a configurar un client, haurem d'instal·lar els paquets `slapd` i `ldap-utils` amb `apt-get install slapd ldap-utils`, que ens sol·licitarà una sèrie d'informació: URI del servidor (`ldap://192.168.1.33`), el DC (`dc=nteum,dc=org`), la versió (3), l'usuari administratiu (`cn=admin,dc=nteum,dc=org`) i el *passwd*. Ens informarà que fem el canvi manual de `nsswitch` (ok), permetem a PAM canviar els *passwords* locals (Yes), *enforces login* (No), *admin account suffix* (`cd=admin,dc=nteum,dc=org`), i finalment el *passwd* novament.

A continuació, hem de modificar l'arxiu `/etc/nsswitch.conf` amb:

```
passwd: compat ldap
group: compat ldap
shadow: compat ldap
...
netgroup: ldap
```

Finalment haurem de modificar l'arxiu `/etc/pam.d/common-password` (traient de la línia `l"use_authok"`)

```
password [success=1 user_unknown=ignore default=die] pam_ldap.so
try_first_pass
```

i en `/etc/pam.d/common-session` agregar al final (per a crear el *home directory* de manera automàtica)

```
session optional pam_mkhomedir.so skel=/etc/skel umask=077.
```


Després, haurem de reiniciar la màquina (`shutdown -r now`) i ja ens podrem connectar amb els usuaris que hem creat (`jpirulo` o `apirulo`) des de la interfície gràfica.

1.13. Serveis d'arxius (NFS, *Network File System*)

El sistema NFS permet a un servidor exportar un sistema d'arxiu perquè pugui ser utilitzat de manera interactiva des d'un client. El servei es compon bàsicament d'un servidor (bàsicament representat per `nfsd*`) i un client (representat per `rpc.mountd`) que permeten compartir un sistema d'arxiu (o part d'aquest) a través de la xarxa. En l'última versió de NFSv4, s'inclouen una sèrie de *daemons* més com `idmapd`, `statd`, a més d'una sèrie de mòduls per a les noves funcionalitats d'aquesta versió. En Debian, instal·leu `apt-get install nfs-common` (serà necessari el paquet `rpcbin` que, com es va comentar abans, és el nou *portmap* i generalment ja està instal·lat) per al client, mentre que el servidor necessita `apt-get install nfs-kernel-server`. El servidor (en Debian) es gestiona mitjançant l'*script* `/etc/init.d/nfs-kernel-server` o simplement amb `service nfs-kernel-server start|stop`. El servidor fa servir un arxiu (`/etc/exports`) per a gestionar l'accés remot als sistemes d'arxiu i el seu control. Sobre el client (o un altre usuari mitjançant `sudo`), el *root* pot muntar el sistema remot a través de l'ordre:

```
mount -t nfs Ipserver:directori-remot directori_local
```

i a partir d'aquest moment, el `directori-remot` es veurà dins del directori local (aquest ha d'existir abans d'executar el `mount`). Aquesta tasca en el client es pot automatitzar utilitzant l'arxiu de *mount* automàtic (`/etc/fstab`) incloent una línia; per exemple:

```
remix.world:/home /home nfs defaults 0 0.
```

Això ens indica que es muntarà el directori `/home` del *host* `remix.world` en el directori local `/home`. A més, aquest sistema d'arxiu es muntarà amb els paràmetres per defecte (vegeu `man mount`, apartat `mount options for ntfs` i `man nfs` per a opcions específiques per a NFSv4). Els últims dos zeros indiquen que el sistema d'arxius no ha de ser *dumped* i que no s'hi activarà el *fsck*. L'arxiu `/etc/exports` serveix d'ACL (llista de control d'accés) dels sistemes d'arxiu que poden ser exportats als clients. Cada línia conté un sistema de fitxers (*filesystem*) per a exportar seguit dels clients que el poden muntar, separats per espais en blanc. A cada client se li pot associar un conjunt d'opcions per a modificar el comportament (consulteu `man exports` per a veure un llista detallada de les opcions). Un exemple d'això podria ser:

Exemple de /etc/exports

```
/          master(rw) trusty(rw,no_root_squash)
/projects   proj*.local.domain(rw)
/usr        .local.domain(ro) @trusted(rw)
/pub        (ro,insecure,all_squash)
/home       195.12.32.2(rw,no_root_squash) www.first.com(ro)
/user       195.12.32.2/24(ro,insecure)
/home       192.168.1.0/24(rw,sync,fsid=0,no_root_squash,no_subtree_check)
```

La primera línia exporta el sistema d'arxius sencer (/) a `master` i `trusty` en mode lectura/escriptura. A més, per a `trusty` no hi ha *uid squashing* (el `root` del client accedirà com a `root` als arxius `root` del servidor, és a dir, els dos `root` són equivalents malgrat ser de màquines diferents i s'utilitzen per a màquines sense disc). La segona i tercera línies mostren exemples de '*' i de *netgroups* (indicats per @). La quarta línia exporta el directori /pub a qualsevol màquina del món, només de lectura, i permet l'accés de clients NFS que no utilitzen un port reservat pel NFS (opció *insecure*) i tot s'executa sota l'usuari *nobody* (opció *all esquaix*). La cinquena línia especifica un client per a la seva IP i en la sisena, s'especifica el mateix però amb màscara de xarxa (/24) i amb opcions entre parèntesis sense espai de separació. Només hi pot haver espai entre els clients habilitats. L'última línia exporta el directori /home a totes les màquines de la xarxa 192.168.0.* en mode sincronitzat, de lectura/escriptura, amb accés del `root` remot, `fsid` és la identificació de sistema d'arxiu i `no_subtree_check` indica que no es farà la verificació de la ruta/arxiu en una petició sobre el servidor.

Dues ordres útils per a treballar amb l'nfs són `l'exportfs` (que mostra i ens permet actualitzar les modificacions que s'hagin fet sobre l'arxiu */etc/exports*) i `nfsiostat/nfsstat`, que ens permetrà obtenir estadístiques de funcionament sobre NFS i observar el seu funcionament.

1.14. Servidor de wiki

Un (o una) **wiki** (del hawaïà *wiki wiki*, "ràpid") és un lloc web col·laboratiu que pot ser editat per diversos usuaris que poden crear, editar, esborrar o modificar el contingut d'una pàgina web, de manera interactiva, fàcil i ràpida; aquestes facilitats fan d'una wiki una eina eficaç per a l'escriptura col·laborativa. La tecnologia wiki permet que pàgines web allotjades en un servidor públic (les pàgines wiki) siguin escrites de manera col·laborativa mitjançant un navegador, utilitzant una notació senzilla per a donar format, crear enllaços, etc. i conservant un historial de canvis que permet recuperar de manera senzilla qualsevol estat anterior de la pàgina. Quan algú edita una pàgina wiki, els canvis apareixen immediatament a la web, sense passar per cap tipus de revisió prèvia. *Wiki* també es pot referir a una col·lecció de pàgines d'hipertext, que qualsevol persona pot visitar i editar (definició de Wikipedia). Deben té el seu wiki

Enllaç d'interès

Per a saber més sobre MoinMoin, podeu visitar la seva pàgina web en: <http://moinmo.in>. En concret, trobareu les instruccions detallades per a instal·lar MoinMoin en: <http://master19.moinmo.in/InstallDocs>.

en <http://wiki.debian.org/> o també Apache en <http://wiki.apache.org/general/> i ambdues estan basades en **MoinMoin**. MoinMoin és una *Python WikiClone* que permet inicialitzar ràpidament la seva pròpia wiki i només es necessiten un servidor de web i el llenguatge Python instal·lat. A la web de MoinMoin, es troben les instruccions detallades per a instal·lar MoinMoin, però hi ha dues maneres principals de fer-ho: instal·lació ràpida i instal·lació de servidor.

1.14.1. Instal·lació ràpida

- 1) Descarregar el paquet des de <http://moinmo.in/moinmoindownload> que serà, per exemple, per a la versió 1.9 `moin-1.9.7.tar.gz`. Si es vol verificar la integritat del paquet, es pot fer `md5sum moin-x.x.x.tar.gz` i verificar que coincideixin el *hash* generat amb aquell que hi ha a la pàgina de descàrrega.
- 2) Desempaquetar MoinMoin `tar xvzf moin-x.x.x.tar.gz`. Això crearà un directori `moin-x.x.x` en el directori actual amb els arxius en el seu interior.
- 3) Com que MoinMoin està escrita en Python, és necessari utilitzar l'interpret de Python:

```
cd moin-x.x.x; python wikiserver.py
```

Aquesta ordre mostrarà per pantalla els missatges d'execució del servidor. Entre aquesta informació es mostrarà l'adreça IP sobre la qual està corrent el servidor, que podrà ser alguna cosa com `http://127.0.0.1:8080`. Aquesta opció fa servir un servidor web intern, serà accessible des de la direcció `http://localhost:8080/` i funcionarà fins que es pressioni `Ctrl-C` en el terminal.

1.14.2. Instal·lació de servidor

MoinMoin és una aplicació WSGI (*Web Server Gateway Interface*) i, per tant, el millor entorn per a executar Moin Moin és un que permeti WSGI com, per exemple, Apache amb `mod_wsgi`. En Debian, podem instal·lar el mòdul instal·lant `apt-get install libapache2-mod-wsgi`.

Instal·lació de MoinMoin

Per a instal·lar MoinMoin, s'ha de descarregar l'última versió i descompactar l'arxiu (per exemple, `tar xvzf moin-1.9.7.tar.gz`) i després fer una `cd moin-1.9.7/` i, a continuació, executar:

```
python setup.py install --force --record=install.log --prefix='/usr/local'
```

Per a fer un test simple:

```
cd /usr/local/share/moin/server
python test.wsgi
```

Enllaç d'interès

Les instruccions per a instal·lar WSGI per a Apache i configurar MoinMoin en aquest cas es poden trobar en la següent adreça:
<http://moinmo.in/HowTo/ApacheWithModWSGI>.

En el navegador, introduir com a URL localhost:8000 i veurem la pàgina de test de WSGI.

Copiar la configuració:

```
cd /usr/local/share/moin
cp server/moin.wsgi .
cp config/wikiconfig.py .
```

Agregar un arxiu en */etc/apache2/conf.d/moin.conf* amb el següent contingut:

```
# MoinMoin WSGI configuration
# you will invoke your moin wiki at the root url, like http://servername/FrontPage:
WSGIScriptAlias / /usr/local/share/moin/moin.wsgi
# create some wsgi daemons - use these parameters for a simple setup
WSGIDaemonProcess moin user=www-data group=www-data processes=5
threads=10
maximum-requests=1000 umask=0007
# use the daemons we defined above to process requests!
WSGIProcessGroup moin
```

Modificar l'arxiu */usr/local/share/moin/moin.wsgi* agregant al final del paràgraf a2: `sys.path.insert(0, '/usr/local/share/moin')`

Modificar els permisos dels directoris/pàgines:

```
cd /usr/local/share; chown -R www-data:www-data moin;
chmod -R ug+rwX moin; chmod -R o-rwx moin
```

Verificar que tinguem un *site* per defecte en Apache (si no, s'ha de fer `a2ensite default`) i reiniciar Apache (`service apache2 restart`)

Si ens connectem a l'URL localhost, tindrem la pàgina inicial de MoinMoin. Per a configurar el nom de la wiki i l'usuari administrador, podem editar l'arxiu */usr/local/share/moin/wikiconfig.py*, traiem el comentari de `page_front_page = o"FrontPage"` i indiquem el nom de l'administrador, p. ex., `superuser = [o"WikiAdmin",]`, i reiniciem Apache novament. Per a configurar el llenguatge, hem d'entrar com a administrador (WikiAdmin) (si no tenim un usuari, seleccionem *login*, seleccionem 'you can create one now' i i el creem; ha de coincidir amb aquell que introduïm com a superuser). Després, podrem configurar l'idioma des de

http://localhost/LanguageSetup?action=language_setup

i a partir d'aquesta acció, ja podrem començar a crear la nostra primera wiki (informació addicional en <http://moinmo.in/howto> i particularment en l'adreça <http://moinmo.in/howto/ubuntuquick>).

Per a configurar múltiples wikis, primer heu de copiar `config/wikifarm/*` de la distribució en el directori `moin/config/`. Després, s'han de seguir les

instruccions anteriors per a cadascuna de les wikis de la col·lecció (*farm*), tenint en compte que:

- 1) és necessari tenir `data_dir` i `data_underlay_dir` separats per a cada wiki,
- 2) si busqueu que comparteixin alguna configuració, llavors aquesta ha d'estar en `farmconfig.py` i les específiques han d'estar en `mywiki.py`.

1.15. Gestió de còpies de seguretat (*backups*)

Les còpies de seguretat (*backup*) es refereixen a una còpia de les dades originals que es fa amb la finalitat de disposar d'un mitjà per a recuperar-les en cas de pèrdua total o parcial a causa de fallades en els dispositius físics, esborrats per accident o atacs informàtics, infectats per virus o altres causes que fan que la informació no existeixi o no sigui la desitjada. El procés de còpia de seguretat es complementa amb un procés de restauració de les dades (*restore*) que pot ser total o parcial/selectiu, que permet retornar el sistema informàtic al punt en el qual es van emmagatzemar les dades. Això pot significar la pèrdua d'informació entre el moment en què es fa la còpia de seguretat i el moment en què es detecta que les dades no existeixen o estan corrompudes, per la qual cosa la política de planificació de còpies de seguretat ha de ser una de les activitats importants en tot administrador de sistemes.

S'ha de tenir en compte que la pèrdua de dades és habitual (i no per això sense conseqüències i, en alguns casos, fatals), ja que, d'acord amb estadístiques recents, més del 60% dels usuaris d'Internet declaren haver patit una seriosa pèrdua de dades en alguna ocasió, i segons un estudi de la Universitat de Texas, només el 6% d'empreses amb pèrdua catastròfica de dades sobreviurà, enfront d'un 43% que mai reobrirà el negoci i un 51% que haurà de tancar en un termini de 2 anys. Per a reafirmar més encara la necessitat de còpies de seguretat, i pel que fa a aquells sistemes que continguin dades de caràcter personal i que estiguin subjectes a la legislació del país (p. ex., a l'Estat espanyol la LOPD, Llei Orgànica de Protecció de Dades), una de les obligacions que han de complir les empreses/institucions/individus és tenir còpies de seguretat per a preservar les dades que tenen emmagatzemades i que estan subjectes a aquesta normativa.

1.15.1. Programes habituals de còpies de seguretat

Hi ha diverses opcions per a fer còpies de seguretat amb diferents objectius, prestacions i interfícies en totes les distribucions GNU/Linux (p. ex., `backintime`, `bacula`, `backup2l`, `backuppc`, `bup`, `chiark`, `dejadup`, `dirvish`, `flexbackup`, `lucky`, `rdiff`, `vbackup`, entre d'altres). Una de les més potents és **Bacula*** (<http://blog.bacula.org/>), que és una col·lecció d'ei-

*<http://www.bacula.org>

nes per a fer còpies de seguretat en una xarxa. Bacula es basa en una arquitectura client/servidor que resulta molt eficaç i fàcil de manejar, ja que presenta un conjunt molt ampli de característiques i és eficient tant per a un conjunt d'ordinadors personals com per a grans instal·lacions. El paquet està format per diferents components, entre els més importants es poden trobar:

- **Bacula-director**, *daemon* que gestiona la lògica dels processos de *backup*.
- **Bacula-storage**, *daemon* encarregat de manejar els dispositius d'emmagatzematge.
- **Bacula-file**, *daemon* per mitjà del qual Bacula obté els fitxers que necessita per a fer la còpia de seguretat i que s'hauran d'instal·lar en les màquines font dels fitxers que cal fer còpia de seguretat, i
- **Bacula-console**, que permet interactuar amb el servei de *backup*.

Bacula suporta discs durs, cintes, DVD, USB i també diferents bases de dades (MySQL, PostgreSQL i SQLite), però com a contrapartida és necessari disposar de tots els paquets instal·lats i la seva instal·lació i posada a punt poden resultar complexes.

Un altre paquet interessant és **BackupPC***, que permet fer còpies de seguretat de disc a disc amb una interfície basada en el web. El servidor s'executa en qualsevol sistema Gnu/Linux i admet diferents protocols perquè els clients puguin escollir la forma de connectar-se al servidor. Aquest programa no és adequat com a sistema de còpia de seguretat d'imatges de disc o particions, ja que no suporta còpies de seguretat en un nivell de bloc de disc; no obstant això, és molt simple de configurar i la possible intrusió sobre la xarxa d'ordinadors en la qual es desitja fer el suport és mínima. Aquest servidor incorpora un client *Server Message Block* (SMB) que es pot utilitzar per a fer còpia de seguretat de recursos compartits de xarxa d'equips que executen Windows.

*<http://backuppc.sourceforge.net/info.html>

La seva instal·lació és senzilla fent, en Debian, `apt-get install backuppc`. Ens indicarà que seleccionem el servidor web (apache2) i ens indicarà l'usuari i *passwd* que ha creat, no obstant això, aquests *passwd*/usuari es poden canviar amb `htpasswd /etc/backuppc/htpasswd backuppc`. Després, en el nostre navegador posem com a URL `http://localhost/backuppc` i amb l'usuari/*passwd* accedirem a la pàgina principal de l'aplicació on ens donarà l'estat del servidor i les opcions.

Hi ha diverses formes de configurar els clients per a fer les còpies de seguretat i dependrà del mètode per a fer-ho i del sistema operatiu. Per a això, cal consultar en l'apartat de documentació com es configurarà cadascuna de les transferències (<http://backuppc.sourceforge.net/faq/BackupPC.html>).

En el cas d'un sistema (remot) Gnu/Linux, des de la interfície d'administració cal editar-ne la configuració (*host* i *xfer*) i confirmar que s'ha definit com a mètode `rsync` i el directori per a fer la còpia. Com que les còpies de seguretat es fan mitjançant `rsync` en combinació amb `ssh`, és necessari que l'usuari

backuppc del servidor pugui accedir com a *root* sense clau a la màquina remota. Per a això, s'ha d'adoptar l'entitat de l'usuari *backuppc* (`su - backuppc`) i generar les claus (sense *passwd*) `ssh-keygen -t dsa` i després utilitzar l'ordre `ssh-copy-id root@client` per a copiar la clau. S'ha de verificar que es pot accedir a l'usuari *root* del client a través de *ssh* i sense *passwd*. A partir d'aquest punt, n'hi haurà prou de seleccionar l'equip remot en què s'ha de fer el *backup* des de la interfície d'administració i iniciar una primera còpia seleccionant el botó *Començar còpia de seguretat completa*.

En sistemes Windows, la manera més simple de portar a terme *backups* és mitjançant el protocol SMB, per la qual cosa sobre el sistema Windows s'haurà d'ingressar com a administrador i configurar el sistema per a compartir carpetes de les quals es vulgui fer la còpia de seguretat o bé el disc dur complet (per exemple, C:), i definir un usuari/*passwd* per a compartir aquest recurs. Des de la interfície d'administració, s'ha d'editar la configuració del host remot amb Windows del qual s'han de fer les còpies de seguretat (per la seva IP, per exemple), l'usuari i el mètode smb (no oblideu fer *Save* després d'haver modificat aquestes dades). Definiu en la pestanya *Xfer* el nom del recurs compartit que cal fer les còpies de seguretat en l'equip remot i el nom de l'usuari i clau d'accés de recurs compartit de l'equip Windows remot. A partir d'aquest punt, n'hi haurà prou de seleccionar l'equip remot des de la interfície d'administració i iniciar un primer suport seleccionant el botó *Començar còpia de seguretat completa*.

Amb Backuppc, es pot definir la freqüència dels suports totals i suports incrementals. De manera predeterminada, el valor per als suports totals és cada 7 dies i per als incrementals és cada dia. Es recomana utilitzar un valor lleugerament inferior als dies. Per exemple, 6,97 en lloc de 7 dies i 0,97 en lloc d'1 dia per a millorar la granularitat del suport (recordeu sempre fer *Save* després de modificar cada opció).

1.15.2. **rdiff-backup i rdiff-backups-fs**

Per a administradors o usuaris avançats, hi ha l'opció de *rdiff-backup*, que és una ordre per a la còpia de seguretat d'un directori a un altre i que també pot ser a través d'una xarxa. El directori de destinació posseirà una còpia del directori font però també informació addicional (diffs) per a gestionar millor les còpies incrementals (encara d'arxius antics). L'aplicació també preserva sub-directoris, enllaços no simbòlics (*hard links*), arxius dev, permisos, propietat (uid/gid), dates de modificació, atributs estesos i ACL i pot operar de manera eficient a través d'un *pipe* a *rsync* per exemple, o utilitzar *rdiff-backup* + *ssh* per a fer *backups* incrementals en lloc remot transmetent només les diferències. També és habitual (i molt simple) tenir un proveïdor de serveis (Gdrive, Dropbox, etc.) amb un directori sobre l'ordinador local que serà sincronitzat sobre el *cloud* del proveïdor i fer la còpia *rdiff-backup* sobre aquest directori perquè després es transmeti al *cloud* del proveïdor. La forma habitual de treball (després d'instal·lar l'ordre) és molt simple: `rdiff-backup font`

destinació i en el cas remot /algun/dir-local a /algun/dir-remot sobre la màquina hostname.org serà

```
rdiff-backup /algun/dir-local hostname.org::/algun/dir-remot
```

però pot ser al revés

```
rdiff-backup user@hostname.org::/remote-dir local-dir
```

i també podrien ser sobre dues màquines remotes

```
rdiff-backup -v5 -print-statistics user1@host1::/source-dir user2@host2::/dest-dir.
```

Per a recuperar un directori local, es fa simplement mitjançant la còpia i si és remot, `rdiff-backup -r now hostname.org::/remote-dir/file local-dir/file` (podeu veure uns quants exemples més en la següent adreça web: <http://www.nongnu.org/rdiff-backup/examples.html>).

Un dels problemes de recuperar la informació prèviament guardada és que accedir als arxius de còpia de seguretat més recent és fàcil (solament s'ha d'introduir el directori de còpia de seguretat), però és complicat si desitgem accedir i navegar a través de les versions anteriors. `rdiff-backup` permet accedir-hi per un conjunt d'instruccions, que requereixen un coneixement precís dels noms d'arxiu i els temps de còpia de seguretat i que pot ser complicat de recordar o seleccionar. `rdiff-backup-fs` (<https://code.google.com/p/rdiff-backup-fs/>) permet crear un sistema d'arxius en espai d'usuari basat en la informació de `rdiff`. Per a això s'utilitza FUSE (*Filesystem in userspace**), que permet que qualsevol usuari pugui muntar el sistema d'arxius guardat per `rdiff` i navegar per cada increment i còpia efectuada.

*<http://fuse.sourceforge.net>

Una alternativa per a les còpies de seguretat d'un sistema remot per a `ssh` és utilitzar `sshfs` (<http://fuse.sourceforge.net/sshfs.html>), que permet l'accés a l'espai d'usuari a un directori remot mostrant-lo localment, per la qual cosa després aplicant `rdiff` és possible fer còpies incrementals d'aquest recurs remot.

1.16. **Public Key Infrastructure (PKI)**

Per PKI (*Public Key Infrastructure*) s'entén un conjunt de maquinari i programari, polítiques i procediments de seguretat que permeten l'execució amb garanties d'operacions com el xifratge, la signatura digital o el no-repudi de transaccions electròniques. El terme PKI s'utilitza per a referir-se tant a l'autoritat de certificació com a la resta de components, si bé de vegades, de manera errònia, s'utilitza per a referir-se a l'ús d'algorismes de clau pública. En una operació en què s'utilitzi PKI intervenen, a més de qui inicia l'acció i el seu destinatari, un conjunt de serveis que donen validesa a l'operació i garanteixen que els certificats implicats són vàlids. Entre aquests podem comptar amb l'autoritat

de certificació, l'autoritat de registre i el sistema de segellament de temps. La infraestructura PKI utilitza procediments basats en operacions criptogràfiques de clau pública, amb algorismes de xifratge ben coneguts, accessibles i segurs i és per això que la seguretat està fortament lligada a la privadesa de la clau privada i les polítiques de seguretat aplicades per a protegir-la. Els principals usos de la PKI són, entre d'altres els següents: autenticació d'usuaris i sistemes (*login*), identificació, xifratge, signatura digital, comunicacions segures i garantia de no-repudi.[21][22][24]

Com s'ha vist anteriorment (per a Apache+SSL), la generació de certificats digitals és extremadament necessària dins de les necessitats habituals dels administradors i usuaris dels sistemes d'informació, per exemple, per a accedir a una pàgina SSL o per a signar un correu electrònic. Aquests certificats es poden obtenir de les entitats certificadores privades com per exemple StartComm (<https://www.startssl.com/>) i Verisign (<http://www.verisign.es/>), que tenen alguns productes gratuïts (per exemple per a signar correus), però, en la majoria, els productes tenen un cost elevat. També podem recórrer a utilitzar GNUPG (<https://www.gnupg.org/>), que és *Open-Source* i per a determinats finalitats és correcte però no per a altres, o bé considerar entitats de certificació institucionals, per exemple a Catalunya IdCat (<http://idcat.cat/>), que ens permetran obtenir certificats de ciutadà (gratuïts) per a signatura, encriptació, autenticació, no repudi però no per a servidors (Idcat sí que pot expedir un altre tipus de certificats però només és per a l'Administració pública i universitats del país). En el present apartat, instal·larem i configurarem una entitat certificadora arrel (i subentitats CA per als diferents dominis de control d'aquesta CA) basada en l'aplicació TinyCA (si bé existeixen altres paquets com OpenCA –<https://pki.openca.org/>– que són més potents i escalables però són més complexos en la seva configuració) que s'adapta molt bé per a petites i mitjanes institucions/empreses. Una entitat certificadora és la base de la infraestructura PKI i emet certificats per a donar garanties d'autenticitat d'una comunicació, un lloc o una informació. Quan instal·lem Apache, hem autosignat el certificat digital (és a dir, nosaltres hem fet tots els passos: petició, generació i signatura) que codifica la comunicació però això no dóna garantia si no es verifica la signatura del certificat autosignat. Per a solucionar aquest problema, i sense recórrer a un proveïdor públic/privat, crearem la nostra pròpia estructura de CA i distribuïrem per canals segurs als nostres usuaris/clients el certificat personal / de servidors i el certificat arrel de la CA perquè els instal·lin en els seus navegadors (com ja ho estan els de StartComm, Verisign, Catcert/Idcat o altres entitats de certificació). Això es fa de manera que quan un lloc web, per exemple, presenti al navegador un certificat digital per a codificar la comunicació SSL, el navegador reconegui el lloc pel certificat arrel que té instal·lat i hi confiï (i no surti la típica finestra d'avertiment de lloc insegur) mantenint la privadesa de les comunicacions. Els usos d'aquests certificats creats per aquesta CA poden ser diversos: per a signar/encriptar un correu, per a validar els nostres servidors SSL o per a configurar la VPN, entre d'altres.

La pràctica habitual és tenir una CA i crear Sub-CA (una per a cada domini) perquè el sistema sigui escalable, per la qual cosa la CA signarà la creació de

noves Sub-CA i després d'aquestes (el seu responsable), tindran capacitat per a signar els seus certificats i totes tindran el mateix certificat arrel de la root-CA (guia molt detallada en [25]). Una vegada instal·lada (`apt-get install tinyca`), executem `tinyca2` i ens presentarà la pantalla principal i indicacions per a crear una nova CA que serà la rootCA. Completeu la pantalla amb les dades, p. ex., *Name=Rootca-nteum.org*, *Data for CA=Rootca-nteum.org*, *SP, passwd, BCN, BCN, NTEUM, NTEUM, adminp@sysdw.nteum.org, 7300, 4096, SHA-1* per a tots els camps. Quan fem OK, apareixerà una nova pantalla per a configurar la rootCA. Cal seleccionar “*Certificate Signing/CRL Signing*”, “*critical*” i en Netscape *Certificate Type*=“*SSL CA, S/MIME CA, Object Signing CA.*”, i si es desitja posar un URL per a revocar els certificats (la qual cosa pot ser una bona idea) es poden emplenar els camps corresponents, després finalment OK. Amb això es crearan els certificats i apareixerà la pantalla principal de tinyCA amb 4 tabuladors on indica CA (informació sobre la CA activa), *Certificates* (mostra els certificats creats per la CA) *Keys* (mostra les claus dels certificats) i *Requests* (peticions de certificats que esperen ser signats per la CA). Per damunt, apareixen una sèrie d'icones per a accedir a funcions directes però Tinyca2 té un error i no mostra els noms de les icones.

El següent pas és crear una nova sub-CA i, per a fer-ho, verificar que estem en el tab de la CA i fer clic en la tercera icona des de la dreta, que ens mostrarà una finestra que com a subtítol té “*Create a new Sub CA*”. Haurem d'introduir el *passwd* que vam posar en la rootCA, donar-hi un nom (subca-sysdw.nteum.org), *Data-for-CA* (sysdw.nteum.org) i la resta de dades com vam fer anteriorment (el *passwd* no ha de ser necessàriament el mateix de la rootCA) i quan fem OK ens sortirà la finestra principal però amb la Sub-CA seleccionada, si volem tornar a la CA haurem d'anar al menú *File->Open* i obrir la rootCA. Recordeu que la Sub-CA serà qui gestionarà les peticions i signarà els certificats per a aquest domini, per la qual cosa hem de seleccionar l'adequada a cada moment. La rootCA només la utilitzarem per a crear/revocar noves sub-CA i per a exportar el certificat arrel de la rootCA (necessari per a enviar-los als nostres clients perquè l'instal·lin en els seus navegadors).

Per a crear un certificat que permeti certificar el nostre *web-server* quan utilitza SSL (<https://sysdw.nteum.org>), amb la nostra Sub-CA seleccionada anem al tab de *Request* i amb el botó dret seleccionem “*New request*” i s'obrirà una finestra on haurem d'introduir l'URL del nostre servidor (sysdw.nteum.org) com a *CommonName* i emplenar la resta de dades. Una vegada creat, el seleccionem i amb el botó dret indiquem “*Sign request*” i seleccionem “*Server request*”, que ens mostrarà una finestra amb el *password* de la CA i la validesa. Aquest procediment és el que genera confiança ja que estem validant la informació aportada en la petició i signant el certificat (que ja podrem veure en els tabs corresponents).

Ara haurem d'exportar els certificats (del web i la rootCA), la *key* i configurar Apache perquè els incorpori. En primer lloc, exportarem la rootCA i la introduïrem en Firefox/Iceweasel. Per a això, en la finestra principal *File->OpenCA* seleccionem la nostra rootCA i seleccionant la segona icona de la

dreta que correspon a “*Export CA Certificate*” indiquem un nom d’arxiu i el format (PEM és l’adequat) i desem el certificat en el nostre sistema d’arxiu (p. ex., /tmp/Rootca-nteum.org.pem). És important tenir en compte fer un `chmod 644 /tmp/Rootca-nteum.org.pem`, ja que es desarà com a 600 i altres usuaris no el podran importar. Des de Firefox/Iceweasel seleccionem *Preferences/Setting->Advanced->Certificates->View Certificates->Authorities->Import* i seleccionem l’arxiu abans creat marcant totes les *trust settings* que ens presenta en la finestra següent (3). Després, podrem veure amb el nom que vam donar a *Organization* el certificat corresponent.

A continuació, en TinyCA2 obrim la nostra sub-CA, seleccionem el tab *Certificates* i sobre el certificat seleccionem el botó dret i indiquem *Export certificate* i donem un nom d’arxiu, per exemple `sysdw.nteum.org-cert.pem`, després repetim el procés amb la *key* en el *Key* tab, fem *Export key* i el desem, per exemple com a `sysdw.nteum.org-key.pem`. És important decidir si posem *passwd* o no, ja que si el posem cada vegada que arrenqui el servidor ens sol·licitarà el *passwd*. Sobre el tab *CA* haurem d’exportar ara el *certificate chain*, que és la primera icona des de la dreta i desar-lo com a `sysdw.nteum.org-chain.pem`. Mourem aquests tres arxius a `/etc/ssl/private/` i passarem a configurar Apache modificant l’arxiu que configuri el nostre lloc SSL, per exemple nosaltres hem utilitzat `/etc/apache2/sites-available/default-ssl`, en el qual hem modificat (només es mostra les línies principals):

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
    ServerName sysdw.nteum.org
    ...
    SSLEngine on
    SSLCertificateFile /etc/ssl/private/sysdw.nteum.org-cert.pem
    SSLCertificateKeyFile /etc/ssl/private/sysdw.nteum.org-key.pem
    SSLCertificateChainFile /etc/ssl/private/sysdw.nteum.org-chain.pem
    ...
</VirtualHost>
</IfModule>
```

Només ens queda habilitar el lloc (`a2ensite default-ssl`) i reiniciar el servidor Apache (`service apatxe2 restart`) (ens demanarà el *passwd* si l’hem posat en la *key*) i provar (en el navegador en què tenim instal·lat el certificat arrel de la rootCA) l’URL `https://sysdw.nteum.org`. Si tot està correcte, carregarà la pàgina sense la típica finestra que informa que el lloc no és segur.[25]

Per a crear certificats per a una adreça de correu i distribuir-los als nostres usuaris juntament amb el certificat de la rootCA, podem fer en el tab *Certificates* de la nostra sub-CA, seleccionar *New - Create Key and Certificate (Client)* i entrar el nom i l’adreça de correu per la qual volem validar, així com el *passwd* per a protegir-lo fins que arribi al seu nou destinatari (i que després el podrà o l’haurà de canviar). A continuació, hem d’exportar-lo tenint en compte utilitzar el format PKCD#12 que inclou el certificat i la clau. Després d’enviar a l’usuari l’arxiu amb el certificat més el de la root-CA, podrà agregar-lo al seu gestor de

correu de manera similar a *root-CA* però com a *personal certificates*. Després, podrà enviar correus signats digitalment i el destinatari (que haurà de tenir instal·lat el certificat de la *rootCA*) podrà verificar la signatura del correu.

Com a anotació final en relació amb la PKI, tots els nostres usuaris/clients/visitants dels nostres serveis hauran de tenir instal·lat el certificat de la *root-CA* en els seus navegadors/clients per a, d'aquesta manera, validar els llocs/serveis que estan sota el nostre domini/subdominis ja que els navegadors/clients de correu no incorporen aquests certificats arrel per defecte. Si el nostre domini/serveis ho requereix, podem gestionar amb Mozilla la inclusió del nostre certificat en <http://www.mozilla.org/en-us/about/governance/policies/security-group/certs/> però no és un tràmit fàcil i és necessari complir amb una sèrie de requisits ja que les garanties del sistema rau en la inclusió d'aquests certificats.

Activitats

1. Configureu un servidor DNS com a cau i amb un domini propi.
2. Configureu un servidor/client NIS amb dues màquines exportant els directoris d'usuari del servidor per NFS.
3. Configureu un servidor SSH per a accedir des d'una altra màquina sense contrasenya.
4. Configureu un servidor Apache + SSL+ PHP+ MySQL+ PHPAdmin per a visualitzar els fulls personals dels usuaris.
5. Configureu un servidor Apache + un Reverse Proxy per a accedir al servidor web des d'una màquina externa a través del Proxy.
6. Creeu i configureu un sistema de correu electrònic a través d'Exim, Fetchmail, SpamAssassin i un servidor IMAP per a rebre correus des de l'exterior i poder llegir-los des d'una màquina remota amb el client Mozilla (Thunderbird).
7. Instal·leu la wiki MoinMoin i creeu un conjunt de pàgines per verificar el seu funcionament.
8. Instal·leu el servidor de *backups* BackupPC i genereu una còpia de seguretat des d'una màquina Windows i una altra des d'una màquina Linux. Escolliu el mètode de comunicació amb els clients i justifiqueu la resposta.
9. Configureu una CA amb *tiny CA* i genereu/proveu els certificats per a una pàgina web amb SSL i per enviar correus signats i verificar-los des d'un altre compte.

Bibliografia

Tots els URL han estat visitats per última vegada el juny del 2014.

- [1] **Debian.org.** *Debian Home.*
<<http://www.debian.org>>
- [2] **Server-World.**
<http://www.server-world.info/en/note?os=Debian_7.0>
- [3] **Langfeldt, N.** *DNS HOWTO.*
<<http://tldp.org/HOWTO/DNS-HOWTO.html>>
- [4] **IETF.** Repositori de Request For Comment desenvolupats per Internet Engineering Task Force (IETF) al Network Information Center (NIC). <<http://www.ietf.org/rfc.html>>
- [5] **Instituto de Tecnologías Educativas.** *Redes de área local: Aplicaciones y Servicios Linux.*
<<http://www.ite.educacion.es/formacion/materiales/85/cd/linux/indice.htm>>
- [6] **Trenholme, S.** *MaraDNS - A small open-source DNS server.*
<<http://maradns.samiam.org/>>
- [7] **DNSMasq** *DNS forwarder and DHCP server.*
<<https://wiki.debian.org/HowTo/dnsmasq>>
- [8] *Comparison of FTP client software*
<http://en.wikipedia.org/wiki/Comparison_of_FTP_client_software>
- [9] *Servicios de red: Postfix, Apache, NFS, Samba, Squid, LDAP*
<<http://debian-handbook.info/browse/es-ES/stable/network-services.html>>
- [10] *NIS HOWTO.*
<<http://www.linux-nis.org/>>
- [11] **Kukuk, T.** *The Linux NIS(YP)/NYS/NIS+ HOWTO -obs:EOL-.*
<<http://tldp.org/HOWTO/NIS-HOWTO/verification.html>>
- [12] *Exim.*
<<http://www.exim.org/docs.html>>
- [13] *ProcMail.*
<<http://www.debian-administration.org/articles/242>>

- [14] **Apache HTTP Server Version 2.2.**
<<http://httpd.apache.org/docs/2.2/>>
- [15] **Apache2 + WebDAV.**
<<http://www.debian-administration.org/articles/285>>
- [16] **Squid Proxy Server.**
<<http://www.squid-cache.org/> >
- [17] **Squid Configuration Examples.**
<<http://wiki.squid-cache.org/ConfigExamples> >
- [18] **Kiracofe, D.** *Transparent Proxy with Linux and Squid mini-HOWTO -obs:EOL però interessant en conceptes-*.
<<http://tldp.org/HOWTO/TransparentProxy.html#toc1>>
- [19] **Pinheiro Malère, L. E.** (2007). *Ldap. The Linux Documentation Project.*
<<http://tldp.org/HOWTO/LDAP-HOWTO/>>
- [20] *Proftpd.*
<<http://www.debian-administration.org/articles/228>>
- [21] **PKI Public-key cryptography.**
<http://en.wikipedia.org/wiki/Public_key_cryptography >
- [22] *SSL/TLS Strong Encryption: An Introduction.*
<http://httpd.apache.org/docs/2.2/ssl/ssl_intro.html#cryptographictech>
- [23] *Transparent Multi-hop SSH.*
<<http://sshmenu.sourceforge.net/articles/transparent-mulithop.html>>
- [24] **Christof Paar, Jan Pelzl** *Introduction to Public-Key Cryptography.*
<<http://wiki.crypto.rub.de/Buch/movies.php> >
- [25] **Magnus Runesson** (2007). *Create your own CA with TinyCA2.*
<<http://theworldofapenguin.blogspot.com.es/2007/06/create-your-own-ca-with-tinyc2-part-1.html>>
- [26] **textsboldGreg Ferro** *Fast Introduction to SOCKS Proxy.*
<<http://etherealmind.com/fast-introduction-to-socks-proxy/> >
- [27] *Proxy SOCKS.*
<<http://en.flossmanuals.net/bypassing-es/proxis-socks/> >
- [28] **Mourani, G.** (2001). *Securing and Optimizing Linux: The Ultimate Solution.* Open Network Architecture, Inc.
<<http://www.tldp.org/LDP/solrhe/Securing-Optimizing-Linux-The-Ultimate-Solution-v2.0.pdf>>
- [29] *Samba.*
<<http://www.samba.org/> >
- [30] *Wiki - Samba.*
<<https://wiki.samba.org> >
- [31] *RSAT. Remote Server Administration Tools on a Windows workstation.*
<https://wiki.samba.org/index.php/Installing_RSAT_on_Windows_for_AD_Management>
- [32] **M. López, C. Alonso** *Samba 4: Controlador Active Directory.*
<<http://waytoit.wordpress.com/2013/05/12/samba-4-controlador-active-directory-parte-1-de-3/> >
- [33] **M. Rushing** *Compiling Samba 4 on Debian Wheezy - Active Directory Domain Controllers.*
<<http://mark.orbum.net/2014/02/22/compiling-samba-4-on-debian-wheezy-active-directory-domain-controllers-ho/> >
- [34] *Guía Samba4 como Controlador de Dominio y Directorio Activo .*
<<http://fraterneo.wordpress.com/2013/08/19/guia-samba4-como-controlador-de-dominio-y-directorio-activo-actualizacion/> >

Administració de dades

Remo Suppi Boldrito

PID_00212469

Índex

Introducció	5
Objectius	7
1. Administració de dades	9
1.1. PostgreSQL	10
1.1.1. Instal·lació de PostgreSQL	11
1.1.2. Com s'ha de crear una base de dades?	12
1.1.3. Com es pot accedir a una base de dades?	13
1.1.4. Usuaris de la base de dades	14
1.1.5. Manteniment	15
1.2. El llenguatge SQL	16
1.2.1. Entorns d'administració gràfics	18
1.3. MySQL	19
1.3.1. Instal·lació de MySQL	21
1.3.2. Postinstal·lació i verificació de MySQL	22
1.3.3. El programa monitor (client) mysql	22
1.3.4. Cas d'ús: processament de dades amb MySQL	23
1.3.5. Administració de MySQL	26
1.3.6. Interfícies gràfiques	27
1.4. MariaDB	28
1.4.1. Instal·lació MariaDB sobre Debian	29
1.5. SQLite	30
1.6. Source Code Control System	31
1.6.1. Concurrent Versions System (CVS)	32
1.6.2. Subversion	36
1.6.3. Git	39
1.6.4. Mercurial	42
1.7. Mantis Bug Tracker	43
Activitats	45
Bibliografia	45

Introducció

Segons afirmen alguns experts, entre el 2013-2018 es generaran tantes dades com en els últims 5000 anys! Aquest fenomen, conegut com a *big data*, és utilitzat per a referir-se a situacions en què el conjunt de dades que cal tractar supera les mesures habituals de dades gestionades avui dia pels sistemes d'informació i hauran de ser obtingudes, emmagatzemades i processades en temps raonables. Com es considera *Big Data*? És un criteri dinàmic, però normalment es pot considerar com un conjunt de dades únic d'entre desenes de *terabytes* i desenes de *petabytes* (10 PBytes = 10.000 TBytes = 10 milions de gigabytes) i en què les dificultats que es trobaran estaran en la seva captura, emmagatzematge, cerca, compartició, lectura, anàlisi i visualització. Vinculades a aquesta nova tendència, han recuperat força tecnologies conegudes com a *data mining* (mineria de dades) en la versió més enfocada al processament de dades socials (*social data mining*) o el *datawarehouse* (repositori de dades), que seran aspectes predominants per a tenir en compte en el futur proper.[1]

Per aquest motiu, la gestió de dades és un aspecte molt important en la utilització de la tecnologia actual, i els sistemes operatius són la base en què s'executaran les aplicacions que capturaran, emmagatzemaran i gestionaran aquestes dades. Un maneig eficient en l'emmagatzematge, gestió i processament de les dades no pot veure's, avui dia, separat d'una **base de dades** (*databases*, DB), i serà el punt crític per a les noves tecnologies que es desenvolupin a partir de les *Big Data*. Una base de dades és un conjunt estructurat de dades que poden organitzar-se de manera simple i eficient per part d'un gestor d'aquesta base. Les bases de dades actuals es denominen *relacionals*, ja que les dades poden emmagatzemar-se en diferents taules que faciliten la seva gestió i administració (exemples d'aquestes són MySQL/MariaDB, PostgreSQL). Per a això, i amb la finalitat d'estandarditzar l'accés a les bases de dades, s'utilitza un llenguatge denominat SQL (*Structured Query Language*), que permet una interacció flexible, ràpida i independent de les aplicacions amb les bases de dades.

També es necessari destacar que comencen, amb el tractament derivat de les *Big Data*, a desenvolupar-se tecnologies per a gestionar dades no estructurades, en allò que es denominen *bases de dades no estructurades* (o *NoSQL* o també *no solament SQL*) i que defineixen sistemes de gestió de bases de dades que difereixen de les tradicionals BD relacionals (RDBMS), en les quals, per exemple, no fan servir SQL com el principal llenguatge de consultes, ja que no es fan cerques exactes. L'operació més típica és la cerca per similitud, les dades emmagatzemades no requereixen estructures fixes com ara taules i sí utilitzen categories com per exemple clau-valor. Les bases de dades NoSQL han crescut amb l'auge de grans companyies d'Internet (Google, Amazon, Twitter i

Facebook, entre d'altres), ja que necessiten trobar formes de tractament de les dades produïdes que amb les RDBMS no poden o és molt complex/ineficient i en què el rendiment/eficiència en el seu processament i les seves propietats de temps real són més importants que la coherència. Exemple d'aquestes bases de dades són Hadoop-Hbase (utilitzada per gairebé totes les grans companyies d'Internet), MongoDB (NoSQL orientat a documents), Elasticsearch (*multitenancy*, *full-text search engine* amb interfície web RESTful i suport per a *schema-free JSON documents*) o Redis (motor de base de dades en memòria, basat en l'emmagatzematge en taules d'*hashes* –clau/valor).

En aquest mòdul es veuran els gestors més importants de bases de dades en entorns GNU/Linux, una breu ressenya a SQL, així com diferents formes de gestionar dades i repositoris dins d'un entorn distribuït i multiusuari.

Objectius

En els materials didàctics d'aquest mòdul trobareu els continguts i les eines procedimentals per aconseguir els objectius següents:

1. Analitzar les maneres d'emmagatzemar dades en forma massiva i ús eficient.
2. Desenvolupar els aspectes essencials de bases de dades i la seva instal·lació i ús.
3. Treballar amb les tècniques de control de versions i analitzar els seus avantatges.
4. Instal·lar i analitzar les diferents eines de gestió de dades i la seva integració per a entorns de desenvolupament.

1. Administració de dades

En l'actualitat, la forma més utilitzada per a accedir a una base de dades és mitjançant una aplicació que executa codi SQL. Per exemple, és molt comú accedir a una DB a través d'una pàgina web que contingui codi PHP o Perl (els més comuns). Quan un client sol·licita una pàgina, s'executa el codi PHP/Perl incrustat a la pàgina, s'accedeix a la DB i es genera la pàgina amb el seu contingut estàtic i el contingut extret de la DB que posteriorment s'envia al client. Dos dels exemples més actuals de bases de dades són els aportats per PostgreSQL i MySQL (o el seu *fork* MariaDB), que seran objecte de la nostra anàlisi.

També veurem alguns aspectes introductoris sobre SQLite, que és un sistema de gestió de bases de dades relacional (i compatible amb ACID) contingut en una biblioteca (escrita en C) i en què, a diferència dels sistemes anteriors (client-servidor), el motor de SQLite no és un procés independent que es comunica amb el programa sinó que SQLite s'enllaça amb el programa i passa a ser-ne part. El programa utilitza la funcionalitat de SQLite mitjançant crides simples a funcions, reduint l'accés a la base de dades i essent molt més eficients. El conjunt de la base de dades (definicions, taules, índexs i les mateixes dades) és guardat en un sol arxiu estàndard a la màquina *host* i la coherència de la BD s'aconsegueix bloquejant tot el fitxer de base de dades al principi de cada transacció.

D'altra banda, quan es treballa en el desenvolupament d'un programari, existeixen altres aspectes relacionats amb les dades, com la validesa i el seu àmbit (sobretot si existeix un conjunt d'usuaris que treballen sobre les mateixes dades). Hi ha diversos paquets per al control de versions (revisions), però l'objectiu de tots és facilitar l'administració de les diferents versions de cada producte desenvolupat al costat de les possibles especialitzacions fetes per a algun client específic. El control de versions es porta a terme per a controlar les diferents versions del codi font. No obstant això, els mateixos conceptes són aplicables a altres àmbits i no només per al codi font sinó també per als documents, imatges, etc. Encara que un sistema de control de versions es pot fer de manera manual, és molt aconsellable disposar d'eines que facilitin aquesta gestió (CVS, Subversion, GIT, Bazaar, Darcs, Mercurial, Monotone, Codeville, RCS, etc.).

En aquest mòdul, veurem CVS (*Control Version System*), Subversion, GIT i Mercurial per a controlar i administrar múltiples revisions d'arxius, automatitzant l'emmagatzematge, la lectura, la identificació i la barreja de diferents revisions. Aquests programes són útils quan un text es revisa freqüentment i inclou codi font, executables, biblioteques, documentació, gràfics, articles i

altres arxius. Finalment, també s'analitzarà una eina per al seguiment d'incidències i errors en entorn de desenvolupament o de projectes anomenat **Mantis**.

La justificació de CVS i Subversion es pot trobar en què CVS és un dels paquets tradicionals més utilitzats i Subversion (també és conegut com a `svn` per ser el nom de l'eina de línia d'ordres) és un programa de control de versions dissenyat de manera específica per a reemplaçar el popular CVS i que soluciona algunes de les seves deficiències. Una característica important de Subversion és que, a diferència de CVS, els arxius amb versions no tenen cadascun un número de revisió independent. Per contra, tot el repositori té un únic número de versió que identifica un estat comú de tots els arxius del repositori en un determinat moment.

A continuació, veurem dos grans entorns en la gestió de repositoris donades les seves prestacions i utilització en grans projectes: GIT i Mercurial. **GIT** és un programari de control de versions dissenyat per Linus Torvalds, basat en l'eficiència i la fiabilitat del manteniment de versions d'aplicacions quan aquestes tenen un gran nombre d'arxius de codi font. **Mercurial** és un sistema de control de versions (bàsicament desenvolupat en Python), i dissenyat per a obtenir el millor rendiment i escalabilitat, amb un desenvolupament completament distribuït (sense necessitat d'un servidor) i que permet una gestió robusta d'arxius (text o binaris) i amb capacitats avançades de ramificació i integració. Finalment, per a completar un mínim cercle d'eines per a compartir i gestionar dades, es presenta una breu descripció de **Mantis Bug Tracker**, que és una eina de gestió d'incidències (*bug tracker*) de codi obert. Aquesta aplicació està escrita en PHP i requereix una base de dades i un servidor web (generalment MySQL i Apache).

1.1. PostgreSQL

En el llenguatge de bases de dades, PostgreSQL utilitza un model client-servidor [3]. Una sessió de PostgreSQL consisteix en una sèrie de programes que cooperen:

- 1) Un procés servidor que gestiona els arxius de la DB accepta connexions dels clients i fa les accions sol·licitades per aquests sobre la DB. El programa servidor és anomenat en PostgreSQL *postmaster*.
- 2) L'aplicació del client (*frontend*) és la que sol·licita les operacions que cal portar a terme en la DB i que poden ser d'allò més variades; per exemple: eines en mode text, gràfiques, servidors de web, etc.

Generalment, el client i el servidor es troben en diferents *hosts* i es comuniquen mitjançant una connexió TCP/IP. El servidor pot acceptar múltiples peticions de diferents clients i activar per a cada nova connexió un procés que

l'atendrà en exclusiva d'una manera transparent per a l'usuari. Hi ha un conjunt de tasques que poden ser dutes a terme per l'usuari o per l'administrador, segons convingui, i que passem a descriure a continuació.

1.1.1. Instal·lació de PostgreSQL

Aquest pas és necessari per als administradors de la DB, ja que dins de les funcions de l'administrador de DB s'inclou la instal·lació del servidor, la inicialització i configuració, l'administració dels usuaris i tasques de manteniment de la DB. La instal·lació de la base de dades es pot fer de dues maneres: mitjançant els binaris de la distribució, la qual cosa no presenta cap dificultat, ja que els *scripts* de distribució fan tots els passos necessaris per a tenir la DB operativa, o a través del codi font, que serà necessari compilar i instal·lar. En el primer cas (*pre-built binary packages*), es poden utilitzar els gestors de paquets o des de línia d'ordres, per exemple, en Debian el `apt-get`. Per al segon cas, es recomana anar sempre a l'origen (o a un repositori mirall de la distribució original). És important tenir en compte que després la instal·lació des del codi font quedarà fora de la DB de programari instal·lat i es perdran els beneficis d'administració de programari que presentin per exemple `apt-cache` o `apt-get`.^[3]

Instal·lació pas a pas

En aquest apartat, s'optarà per la instal·lació i configuració dels paquets de la distribució (tant de la versió distribuïda com la de l'última versió de desenvolupador), però la instal·lació des de les fonts no genera cap dificultat i és l'adequada si es desitja tenir l'última versió de la BD disponible. Les fonts es poden obtenir des de <http://www.postgresql.org/ftp/source/> i seguint les indicacions des de <http://www.postgresql.org/docs/9.3/interactive/installation.html>.

La instal·lació és molt simple, executant `apt-get install postgresql install` la versió 9.1 de la distribució de Debian Wheezy (a més d'altres paquets addicionals com `postgresql-client-9.1 postgresql-client-common postgresql-common`). En el cas que es desitgi instal·lar l'última versió (9.3) des dels paquets binaris, PostgreSQL facilita els paquets per a Debian* i s'ha d'agregar el repositori, p. ex., creant un arxiu `/etc/apt/sources.list.d/pgdg.list` que tingui la següent línia:

*<http://www.postgresql.org/download/linux/debian/>

```
deb http://apt.postgresql.org/pub/repos/apt/ wheezy-pgdg main
```

Després hem d'importar la clau del repositori amb

```
wget -quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc  
| apt-key add -
```

i actualitzar el repositori `apt-get update`, finalment podrem executar l'ordre `apt-get install postgresql-9.3`. Si es desitja instal·lar el paquet que inclou diferents contribucions de la comunitat*, llavors haurem de fer `apt-get install postgresql-contrib` (aquestes contribucions inclouen *fuzzystrmatch*, *unaccent* o *citext* per a cerques intensives o difuses, per exemple). Després de la instal·lació, tindrem client `postgresql (psql)`, usuari del sistema per defecte. Debian ha creat l'usuari **postgres** (no canvieu el *passwd* d'aquest usuari), usuari per defecte de `postgresql`, és el *superuser postgres* (el seu *passwd* haurà de ser canviat des de dins de la BD); usuari de la BD **postgres**, clúster per defecte, **main**; BD per defecte, **template1**; *schema* per defecte, **public**. Per a canviar el *passwd* de l'usuari **postgres** de la BD fem: `su -l root` a continuació `su -postgres` i després `psql` per a entrar en la consola (client o *frontend*). Després dels missatges i davant del *prompt postgres=#* introduïm `\password postgres` i el *passwd* seleccionat dues vegades. Per a sortir fem `\q`. A continuació, s'hauran de fer alguns ajustos com el de l'arxiu `/etc/postgresql/9.1/main/postgresql.conf` i fer el canvi de la línia `#listen_addresses = 'localhost'` per `listen_addresses = 'localhost, 192.168.1.200'` en què la IP és la de servidor. També caldrà definir el mètode d'autenticació en `/etc/postgresql/9.1/main/pg_hba.conf` i agregar, per exemple

*<http://www.postgresql.org/docs/9.3/static/contrib.html>

```
host all all 192.168.1.200/24 md5
host all all 192.168.1.100/24 md5
```

On 192.168.1.200 és l'adreça de servidor i 192.168.1.100 la IP de gestió de la DB i haurem d'activar, si ho tenim instal·lat, `iptables` per a acceptar comunicacions en el port 5432. Finalment, haurem de reiniciar el servidor amb `service postgresql restart` perquè els canvis siguin efectius. Els arxius essencials de PostgreSQL es troben a: configuració `/etc/postgresql/9.1/main/`, binaris `/usr/lib/postgresql/9.1/bin`, arxius de dades `/var/lib/postgresql/9.1/main` i logs `/var/log/postgresql/postgresql-9.1-main.log`. Per a veure els clústers disponibles (PostgreSQL es basa en CLUSTERS que és un grup de BD manejades per un servei comú i on cadascuna d'aquestes BD conté un o més SCHEMATA), podem executar `pg_lsclusters` que en el nostre cas és:

```
Version Cluster Port Status Owner Data directory Log file
9.1 main 5432 online postgres /var/lib/postgresql/9.1/main /var/log/postgresql/postgresql-9.1-main.log
```

Vegeu la documentació de PostgreSQL en la següent pàgina del seu lloc web: <http://www.postgresql.org/docs/9.1/interactive/index.html>.

1.1.2. Com s'ha de crear una base de dades?

La primera acció per a verificar si es pot accedir al servidor de DB és crear una base de dades. El servidor PostgreSQL pot gestionar moltes DB i és recomanable utilitzar-ne una de diferent per a cada projecte.

Per a crear una base de dades, s'utilitza l'ordre `createdb` des de la línia d'ordres del sistema operatiu. Aquesta ordre generarà un missatge `CREATE DATABASE` si tot és correcte. Amb l'ordre `!l` ens farà una llista de totes les BD definides.

És important tenir en compte que per a dur a terme aquesta acció, hi ha d'haver un usuari habilitat per a crear una base de dades, com hem vist en l'apartat anterior, en què existeix un usuari que instal·la la base de dades i que tindrà permisos per a crear bases de dades i crear nous usuaris que, al seu torn, puguin crear bases de dades. Generalment (i en Debian), aquest usuari és **postgres** per defecte. Per això, abans de fer el `createdb`, s'ha de fer un `su postgres` (o prèviament fer `su -l root`) i a continuació es podrà portar a terme el `createdb`. Per a crear una DB denominada *nteumdb*:

```
createdb nteumdb
```

Si l'execució de l'ordre dóna error, és possible que no estigui ben configurat el camí o que la DB estigui mal instal·lada. Es pot intentar amb el camí absolut (`/usr/lib/postgresql/9.1/bin/createdb nteumdb`), on la ruta dependrà de la instal·lació que s'hagi fet (consulteu referències per a la solució de problemes). Altres missatges d'error serien *could not connect to server*, quan el servidor no està arrencat, o *CREATE DATABASE: permission denied*, quan no es tenen privilegis per a crear la DB. Per a eliminar la base de dades, es pot utilitzar `dropdb nteumdb`.

1.1.3. Com es pot accedir a una base de dades?

Una vegada creada la DB, s'hi pot accedir de diverses formes:

- 1) executant una ordre interactiva anomenada `psql`, que permet editar i executar ordres SQL (per exemple, `psql nteumdb`);
- 2) executant una interfície gràfica com `PhpPgAdmin` o alguna *suite* que tingui suport ODBC per a crear i manipular DB;
- 3) escrivint una aplicació amb alguns dels llenguatges suportats, com PHP, Perl o Java, entre d'altres (consulteu *PostgreSQL Programmer's Guide*).

Per simplicitat, utilitzarem `psql` per a accedir a la DB, per la qual cosa s'haurà d'introduir `psql nteumdb`: sortiran uns missatges amb la versió i informació i un *prompt* similar a `nteumdb =>` o `nteumdb =>` (el primer si és el superusuari i el segon si és un usuari normal). Es poden executar algunes de les ordres SQL següents:

```
SELECT version(); o també SELECT current_date;
```

Nota

Per a poder accedir a la DB, el servidor de base de dades haurà d'estar en funcionament. Quan s'instal·la PostgreSQL, es creen els enllaços adequats perquè el servidor s'iniciï en l'arrencada de l'ordinador. Per a més detalls, consulteu l'apartat d'instal·lació.

`psql` també té ordres que no són SQL i comencen per `\`, per exemple `\h` (enumera totes les ordres disponibles) o `\q` per a acabar. Podem consultar en <http://www.postgresql.org/docs/9.1/static/sql-commands.html> una llista de les ordres SQL que permet.

1.1.4. Usuaris de la base de dades

Els usuaris de la DB són completament diferents dels usuaris del sistema operatiu. En alguns casos, podria ser interessant mantenir una correspondència, però no és necessari. Els usuaris són per a totes les DB que controla aquest servidor, no per a cada DB.

Per a crear un usuari, es pot executar la sentència SQL `CREATE USER nom` i per a esborrar usuaris, `DROP USER nom`.

També es poden cridar els programes `createuser` i `dropuser` des de la línia d'ordres. Hi ha un usuari per defecte anomenat **postgres** (dins de la DB), que és el que permetrà crear els restants (per a crear nous usuaris, si l'usuari de sistema operatiu amb el qual s'administra la DB no és `postgres psql -O usuari`).

Un usuari de DB pot tenir un conjunt d'atributs en funció d'allò que pot fer:

- **Superusuari:** aquest usuari no té cap restricció. Per exemple, podrà crear nous usuaris: `CREATE USER nom SUPERUSER;`
- **Creador de DB:** té permís per a crear DB. Per a crear un usuari d'aquestes característiques, utilitzeu l'ordre: `CREATE USER nom CREATEDB;`
- **Contrasenya:** només és necessari si per qüestions de seguretat es desitja controlar l'accés dels usuaris quan es connectin a una DB. Per a crear un usuari amb contrasenya (`paraula_clau` serà la clau per a aquest usuari): `CREATE USER nom WITH PASSWORD 'paraula_clau';`

A un usuari se li poden canviar els atributs utilitzant l'ordre `ALTER USER`.

També es poden fer grups d'usuaris que comparteixin els mateixos privilegis amb:

```
CREATE GROUP NomGrup;
```

Per a inserir usuaris en aquest grup:

```
ALTER GROUP NomGrup ADD USER Nom1;
```

Per a esborrar:

```
ALTER GROUP NomGrup DROP USER Nom1;
```

Exemple: operacions amb grup dins de `psql`

```
CREATE GROUP NomGrup;  
ALTER GROUP NomGrup ADD USER Nom1,...; ALTER GROUP NomGrup DROP USER  
Nom1,...;
```

Quan es crea una DB, els privilegis són per a l'usuari que la crea (i per al *superusuari*). Per a permetre que un altre usuari utilitzi aquesta DB o una part, se li han de concedir privilegis. Hi ha diferents tipus de privilegis, com `SELECT`, `INSERT`, `UPDATE`, `DELETE`, `RULE`, `REFERENCES`, `TRIGGER`, `CREATE`, `TEMPORARY`, `EXECUTE`, `USAGE` i `ALL PRIVILEGES` (s'han de consultar les referències per a veure el seu significat).

Per a assignar els privilegis, es pot utilitzar `GRANT UPDATE ON objecte TO usuari` on usuari haurà de ser un usuari vàlid de PostgreSQL i objecte, una taula, per exemple.

Aquesta ordre l'haurà d'executar el superusuari o el propietari de la taula. L'usuari `PUBLIC` pot ser utilitzat com a sinònim de tots els usuaris i `ALL` com a sinònim de tots els privilegis. Per exemple, per a treure tots els privilegis a tots els usuaris d'objecte, es pot executar:

```
REVOKE ALL ON objecte FROM PUBLIC;
```

1.1.5. Manteniment

Hi ha un conjunt de tasques que són responsabilitat de l'administrador de DB i que s'han de portar a terme periòdicament:

1) Recuperar l'espai: per a això s'haurà d'executar periòdicament la instrucció `VACUUM`, que recuperarà l'espai de disc de files esborrades o modificades, actualitzarà les estadístiques utilitzades pel planificador de PostgreSQL i millorarà les condicions d'accés.

2) Reindexar: en certs casos, PostgreSQL pot donar alguns problemes amb la reutilització dels índexs, per això, és convenient utilitzar `REINDEX` periòdicament per a eliminar pàgines i files. També es pot utilitzar `contrib/reindexdb` per a reindexar una DB sencera (s'ha de tenir en compte que depenent de la grandària de les DB, aquestes ordres poden trigar un cert temps).

3) Canvi d'arxius de registre (*logs*): s'ha d'evitar que els arxius de registre siguin molt grans i difícils de manejar. Es pot fer fàcilment quan s'inicia el servi-

dor amb `pg_ctl start | logrotate`, en què aquesta última ordre renombra i obre un nou arxiu de registre i es pot configurar amb `/etc/logrotate.conf`.

4) Còpia de seguretat i recuperació (*backup* i *recovery*): hi ha dues formes de guardar les dades, amb la sentència SQL `dump` o guardant l'arxiu de la DB. El primer és `pg_dump ArxiuDB > ArxiuBackup`. Per a recuperar, es pot utilitzar `psql ArxiuDB < ArxiuBackup`. Per a guardar totes les DB del servidor, es pot executar `pg_dumpall > ArxiuBackupTotal`. Una altra estratègia és guardar els arxius de les bases de dades en un nivell del sistema operatiu, per exemple amb `tar -cf backup.tar /var/lib/postgresql/9.1/main`. Hi ha dues restriccions que poden fer que aquest mètode sigui poc pràctic:

- a) el servidor s'ha d'aturar abans de guardar i de recuperar les dades i
- b) s'han de conèixer molt bé totes les implicacions en un àmbit d'arxiu, on hi ha totes les taules, transaccions i altres, ja que en cas contrari, una DB pot quedar inútil. A més, en general, la grandària que es guardarà serà major que allò fet amb els mètodes anteriors, ja que, per exemple, amb el `pg_dump` no es guarden els índexs, sinó l'ordre per a recrear-los.

1.2. El llenguatge SQL

No és la finalitat d'aquest subapartat fer un tutorial sobre SQL, però s'analitzaran uns exemples per a veure les capacitats d'aquest llenguatge. Són exemples que vénen amb la pròpia distribució de les fonts de PostgreSQL en el directori `DirectorioInstal·lació/src/tutorial`. Per a accedir-hi, canvieu al directori de PostgreSQL (`cd DirectorioInstal·lació/src/tutorial`) i executeu `psql -s nteumdb` i dins `\i basics.sql`, després. El paràmetre `\i` llegeix les ordres de l'arxiu especificat (`basic.sql` en el nostre cas). PostgreSQL, com ja hem esmentat, és una base de dades relacional (*Relational Database Management System, RDBMS*), la qual cosa significa que maneja les dades emmagatzemades en taules. Cada taula té un nombre determinat de files i de columnes, i cada columna té un tipus específic de dades. La taules s'agrupen en una DB i un únic servidor maneja aquesta col·lecció de DB (tot el conjunt es denomina *agrupació* o *clúster de bases de dades, database cluster*). Per a crear, per exemple, una taula amb `psql`, executeu:

```
CREATE TABLE temps (  
    ciutat varchar(80),  
    temp_min int,  
    temp_max int,  
    pluja real,  
    dia date  
);
```

L'ordre acaba quan es posa `;` i es poden utilitzar espais en blanc i tabulacions lliurement. `Varchar(80)` especifica una estructura de dades que pot emmagatzemar fins a 80 caràcters (en el nostre cas). El *point* és un tipus específic de PostgreSQL.

Per a esborrar la taula:

```
DROP TABLE nom_taula;
```

Per a introduir dades, es poden utilitzar dues formes: posar totes les dades de la taula o indicar les variables i els valors que es desitgen modificar:

```
INSERT INTO temps VALUES ('Barcelona', 16, 37, 0.25, '2007-03-19');
INSERT INTO temp (ciutat, temp_min, temp_max, pluja, dia) VALUES
('Barcelona', 16, 37, 0.25, '2007-03-19');
```

Aquesta manera pot ser senzilla per a unes poques dades, però quan cal introduir gran quantitat de dades, es poden copiar des d'un arxiu amb la sentència:

`COPY temps FROM '/home/user/temps.txt';` Aquest arxiu ha d'estar en el servidor, no en el client). Per a mirar una taula, podríem fer:

```
SELECT * FROM temps;
```

on el `*` significa "totes les columnes".

Exemple: introduir dades en taula. Dins de `psql`:

```
INSERT INTO NomTB (valorVar1, ValorVar2,...);
```

Dades des d'un arxiu. Dins de `psql`:

```
COPY NomTB FROM 'NomArxiu';
```

Visualitzar dades. Dintre de `psql`:

```
SELECT * FROM NomTB;
```

Alguns exemples d'ordres més complexos serien (dins de `psql`). Visualitza la columna ciutat després de portar a terme l'operació:

```
SELECT ciutat, (temp_max+temp_min)/2 AS temp_media, date FROM temps;
```

Visualitza tot on es compleix l'operació lògica:

```
SELECT * FROM temps WHERE city = 'Barcelona' AND pluja > 0.0;
```

Unió de taules:

```
SELECT * FROM temps, ciutat WHERE ciutat = nom;
```

Funcions, màxim en aquest cas:

```
SELECT max(temp_min) FROM temps;
```

Funcions imbricades:

```
SELECT ciutat FROM temps WHERE temp_min = (SELECT max(temp_min) FROM
temps);
```

Modificació selectiva:

```
UPDATE temps SET temp_max = temp_max/2, temp_min = temp_min/2 WHERE
dia > '19990128';
```

Esborrament del registre:

```
DELETE FROM temps WHERE ciutat = 'Sabadell';
```

Altres ordres d'utilitat:

Login com a usuari postgres (SuperUser) per a treballar amb la base de dades:

```
# su - postgres
```

Crear base de dades: `$ createdb nom_BD`

Eliminar base de dades: `$ dropdb nom_BD`

Accedir a la base de dades: `$ psql nom_BD`

Ajuda (dins de la base de dades; per exemple, `mynteam: mynteam=# \h`

Sortir del `psql`: `mynteam=# \q`

Guardar la base de dades: `$ pg_dump mynteam > db.out`

Carregar la base de dades guardada anteriorment: `$ psql -d mynteam -f db.out`

Per a guardar totes les bases de dades: `# su - postgres i després $ pg_dumpall > /var/lib/postgresql/backups/dumpall.sql`

Per a restaurar una base de dades: `# su - postgres i després`

```
$ psql -f /var/lib/postgresql/backups/dumpall.sql mynteam
```

Per a mostrar les bases de dades: `psql -l` o si no, des de dintre `mynteam=# \l`;

Mostrar els usuaris: des de dins de l'ordre `psql` executar `mynteam=# SELECT * FROM "pg_user";`

Mostrar les taules: des de dins de l'ordre `psql` executar `mynteam=# SELECT * FROM "pg_tables";`

Canviar la contrasenya: des de dins de l'ordre `psql` `mynteam=# UPDATE pg_shadow SET passwd = 'new_password' where username = 'username';` o també

```
ALTER USER nom WITH PASSWORD 'password';
```

Netejar totes les bases de dades: `$ vacuumdb - -quiet - -all`

Per a un usuari que s'inicia en PostgreSQL és interessant obtenir una base de dades ja construïda i practicar amb aquesta. Per exemple, en l'adreça <http://pgfoundry.org/projects/dbsamples/> podem obtenir la base World 1.0, que ens dona una llista de 4.079 ciutats del món de 239 països i la seva població. Una vegada descarregada (format tar.gz) la descomprimim amb `tar xzvf world-1.0.tar.gz` i entrem en `psql`, des del prompt fem

```
\i /tmp/world.sql
```


Veurem com es creen les taules i ja estarem en condicions de fer consultes com:

```
SELECT * FROM "pg_tables" WHERE schemaname = 'public';           --Mirem les taules
schemaname |      tablename      | tableowner | tablespace | hasindexes | hasrules | hastriggers
-----+-----+-----+-----+-----+-----+-----
public    | city                | postgres  |            | t          | f        | t
public    | country             | postgres  |            | t          | f        | t
public    | countrylanguage     | postgres  |            | t          | f        | t

SELECT * FROM "city" WHERE countrycode = 'ESP';                 --Mirem les ciutats d'un país
id |      name      | countrycode |      district      | population
---+-----+-----+-----+-----
653 | Madrid         | ESP         | Madrid             | 2879052
654 | Barcelona      | ESP         | Katalonia          | 1503451
655 | València       | ESP         | València           | 739412
...

SELECT count(*) FROM "city" WHERE countrycode = 'ESP';          --Comptem les ciutats d'un país
count
-----
    59

SELECT * FROM "city" WHERE name = 'Barcelona';                  --Mirem les ciutats amb un nom
id |      name      | countrycode |      district      | population
---+-----+-----+-----+-----
654 | Barcelona      | ESP         | Katalonia          | 1503451
3546 | Barcelona      | VEN         | Anzoátegui         | 322267

SELECT * FROM "city" WHERE name = 'Barcelona' and countrycode = 'ESP'; --I només les d'un país
id |      name      | countrycode |      district      | population
---+-----+-----+-----+-----
654 | Barcelona      | ESP         | Katalonia          | 1503451

SELECT * FROM "city" WHERE population > '9500000';              --Mirem les ciutats amb una població major
id |      name      | countrycode |      district      | population
---+-----+-----+-----+-----
206 | Sao Paulo      | BRA         | Sao Paulo          | 9968485
939 | Jakarta        | IDN         | Jakarta Raya       | 9604900
1024 | Mumbai (Bombay) | IND         | Maharashtra        | 10500000
1890 | Shanghai       | CHN         | Shanghai           | 9696300
2331 | Seoul          | KOR         | Seoul              | 9981619

SELECT TRUNC(AVG(population),1) AS Total FROM "city" WHERE population > '9500000';
--Obtenim la mitjana de les ciutats més poblades
total
-----
9950260.8
```

Per a més detalls sobre PostgreSQL i la seva utilització, recomanem [2][4].

1.2.1. Entorns d'administració gràfics

Hi ha diversos entorns d'administració gràfics, com **Phppgadmin** i **Pgadmin** (disponible en la majoria de distribucions, Debian inclosa). Aquests programes permeten accedir i administrar una base de dades amb una interfície gràfica ja sigui mitjançant web o per interfície pròpia. En el cas d'interfície pròpia, la manera més fàcil d'accedir és des d'una terminal. L'administrador de la DB (si no és l'usuari postgresql) haurà de fer `xhost +`, la qual cosa permet que altres aplicacions puguin connectar-se al *display* de l'usuari actual i, a continuació, executar el nom de l'aplicació.

Per a instal·lar el paquet PhpPgAdmin haurem d'executar `apt-get install phppgadmin` i estarà disponible en l'URL <http://localhost/phppgadmin>. Per tenir-lo disponible des de l'adreça del propi servidor (o des d'una altra), haurem de modificar l'arxiu `/etc/apache2/conf.d/phppgadmin` i comentar la línia `allow from 127.0.0.0/255.0.0.0 ::1/128` i descomentar `allow from all` reiniciant el servidor després. Quan desitgem entrar en la base de dades en el menú de l'esquerra, ens demanarà un usuari/*passwd* que si introduïm "postgres" i el *passwd* del superusuari ens donarà un error dient *Login disallowed for security reasons*. Per a habilitar el seu accés, haurem d'anar a l'arxiu `/etc/phppgadmin/config.inc.php` i llavors haurem de canviar la línia `$conf['extra_login_security'] = true;` per aquesta altra línia:

```
$conf['extra_login_security'] = false;
```

A continuació, haurem de recarregar la pàgina i reiterar el procediment i ja podrem accedir a la gestió de la base de dades.

Si es desitja evitar treballar amb l'usuari "postgres" (és recomanable), es pot crear un usuari amb contrasenya per a accedir a la base de dades, i per a això cal fer des de dins de

```
psql: CREATE USER admin WITH PASSWORD 'posar_passwd';
```

Per veure que tot és correcte, es pot fer `SELECT * FROM "pg_user";` i es veurà l'usuari creat amb * en la contrasenya. En Phppgadmin podrem seleccionar a l'esquerra el servidor i configurar els paràmetres indicant que el servidor és *localhost* i l'usuari i la contrasenya creats anteriorment. A partir d'aquí, veurem les bases de dades, així com tots els paràmetres i configuracions.

Una altra eina interessant és **Webmin**. És una interfície basada en el web per a administrar sistemes per a Unix que permet configurar usuaris, Apache, DNS, compartir arxius, servidors de bases de dades, etc. L'avantatge de Webmin, sobretot per a usuaris que s'inicien en l'administració, és que elimina la necessitat d'editar manualment els arxius de configuració i permet administrar un sistema de forma local o remota. La instal·lació per a Debian es detalla en <http://www.webmin.com/deb.html> (Webmin va ser retirat del repositori Debian/Ubuntu el 2006 per retards en la solució d'errors i problemes en la política de manteniment del paquet <https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=343897>).

Enllaços d'interès

Webmin no està disponible en algunes distribucions i s'ha de baixar i instal·lar directament de la web: <http://www.webmin.com>. Tota la documentació i la informació dels mòduls de Webmin disponibles es troba en: <http://doxfer.webmin.com/Webmin>.

1.3. MySQL

MySQL [7] és, segons els seus autors, la base de dades (DB) SQL oberta, és a dir, de programari lliure (*Open Source*) més popular i utilitzada per grans companyies d'Internet (Facebook, Google, Adobe, Twitter, YouTube, Zappos, entre d'altres). La companyia que desenvolupa i manté la BD i les seves utilitats es

Enllaç d'interès

Tota la documentació sobre MySQL es pot obtenir des de la pàgina web següent: http://dev.mysql.com/usingmysql/get_started.html.

denomina MySQL AB (el 2008, aquesta va ser adquirida per Sun Microsystems i al seu torn aquesta ho va ser el 2010 per Oracle Corporation, per la qual cosa MySQL és subsidiària d'Oracle Co.) i desenvolupa MySQL com a programari lliure en un esquema de llicència dual. D'una banda, s'ofereix sota la GNU GPL per a qualsevol ús compatible amb aquesta llicència (versió anomenada *MySQL Community Edition*), però per a aquelles empreses que vulguin incorporar-ho en productes privats, han de comprar una llicència específica que els permeti aquest ús (versions *MySQL Enterprise Edition* i *MySQL Cluster CGE* bàsicament). És interessant analitzar les diferents situacions (sobretot en la nostra orientació cap a l'*Open Source*) de llicències de MySQL. En l'opció GPL, podem utilitzar lliurement MySQL, és a dir, sense cost quan l'aplicació es desenvolupa en un nivell local i no s'utilitza comercialment. Per exemple, MySQL es pot fer servir lliurement dins d'un lloc web (si es desenvolupa una aplicació PHP i s'instal·la en un proveïdor de serveis d'Internet, tampoc s'ha de fer que el codi PHP estigui disponible lliurement en el sentit de la GPL). Igualment, un proveïdor de serveis d'Internet pot fer que MySQL estigui disponible per als seus clients sense haver de pagar llicència de MySQL (sempre que MySQL s'executi de manera exclusiva en l'equip de l'ISP) o MySQL es pot utilitzar de manera gratuïta per a tots els projectes GPL o llicència lliure comparable (per exemple, si hem desenvolupat un client de correu electrònic *Open Source* per a GNU/Linux, i es desitja emmagatzemar dades dels correus en una base de dades MySQL) i tots els desenvolupaments/extensions que es facin sobre els productes MySQL han de ser oberts/publicats i no comercials. L'ús de MySQL amb una llicència comercial s'aplica quan desenvolupem productes comercials i no estan oberts/publicats; és a dir, no es pot desenvolupar un producte comercial (p. ex., un programa de comptabilitat amb MySQL com a base de dades) sense que el codi estigui disponible com a codi obert. Si les limitacions de la GPL no són acceptables per a qui desenvolupa l'aplicació, llavors es pot vendre el producte (programa), juntament amb una llicència de MySQL comercial (més informació en <http://www.mysql.com/about/legal/>). Considerant la llicència GPL, es pot obtenir el codi font, estudiar-lo i modificar-lo d'acord amb les seves necessitats sense cap pagament, però estarem subjectes als deures que imposa GPL i hauré de publicar com a codi obert el que hem desenvolupat també. MySQL proveeix la seva pàgina web d'un conjunt d'estadístiques i prestacions en comparació a altres DB per a mostrar a l'usuari com de ràpida, fiable i fàcil és d'usar. La decisió de triar una DB s'ha de fer acuradament en funció de les necessitats dels usuaris i de l'entorn on s'utilitzarà aquesta DB.

Com PostgreSQL, MySQL és una base de dades relacional, és a dir, que emmagatzema les dades en taules en lloc d'en una única ubicació, la qual cosa permet augmentar la velocitat i la flexibilitat.

1.3.1. Instal·lació de MySQL

MySQL es pot obtenir des de la pàgina web* del projecte o des de qualsevol dels repositoris de programari. Es poden obtenir els binaris i els arxius font per a compilar-los i instal·lar-los. En el cas dels binaris, s'ha d'utilitzar la distribució de Debian i seleccionar els paquets `mysql-*` (`client-5.5`, `server-5.5` i `common` són necessaris). La instal·lació, després d'unes preguntes, crearà un usuari que serà el superusuari de la base de dades (p. ex., `admin`) i una entrada en `/etc/init.d/mysql` per a arrencar o aturar el servidor en el *boot*. També es pot fer manualment:

*<http://www.mysql.com>

```
/etc/init.d/mysql start|stop
```

Per a accedir a la base de dades, es pot utilitzar el monitor `mysql` des de la línia d'ordres. Si obté els binaris (que no siguin Debian ni RPM, per a aquests simplement utilitzeu les utilitats comunes `apt-get` i `rpm`), per exemple un arxiu comprimit des del lloc web de MySQL, haurà d'executar les següents ordres per a instal·lar la DB:

```
groupadd mysql
useradd -g mysql mysql
cd /usr/local
gunzip < /path/to/mysql-VERSION-OS.tar.gz | tar xvf -
ln -s full-path-to-mysql-VERSION-OS mysql
cd mysql
scripts/mysql_install_db --user=mysql
chown -R root .
chown -R mysql data
chgrp -R mysql .
bin/mysqld_safe --user=mysql &
```

Això crea l'usuari/grup/directori, descomprimeix i instal·la la DB en el directori `/usr/local/mysql`. En cas d'obtenir el codi font, els passos són similars:

```
groupadd mysql
useradd -g mysql mysql
gunzip < mysql-VERSION.tar.gz | tar -xvf -
cd mysql-VERSION
./configure --prefix=/usr/local/mysql
make
make install
cp support-files/my-medium.cnf /etc/my.cnf
cd /usr/local/mysql
bin/mysql_install_db --user=mysql
chown -R root .
chown -R mysql var
chgrp -R mysql .
bin/mysqld_safe --user=mysql &
```

És important parar esment quan es fa la configuració, ja que

```
prefix=/usr/local/mysql
```

és el directori on s'instal·larà la DB i es pot canviar per a situar la DB en el directori que es desitgi.

1.3.2. Postinstal·lació i verificació de MySQL

Una vegada feta la instal·lació (ja sigui dels binaris o del codi font), s'haurà de verificar si el servidor funciona correctament. En Debian es pot fer directament (l'usuari és el que s'ha definit durant la instal·lació igual que la seva contrasenya, indicats en les ordres per `-u` i `-p`, de manera respectiva):

```
/etc/init.d/mysql start    Inicia el servidor
mysqladmin -u user -p version    Genera informació de versions
mysqladmin -u user -p variables    Mostra els valors de les variables
mysqladmin -u root -p shutdown    Finalitza l'execució del servidor
mysqlshow -u user -p    Mostra les DB predefinides
mysqlshow mysql -u user -p    Mostra les taules de la DB mysql
```

Si s'instal·la des del codi font, abans de fer aquestes comprovacions s'han d'executar les següents ordres per a crear les bases de dades (des del directori de la distribució):

```
./scripts/mysql_install_db
cd DirectorioInstalacionMysql
./bin/mysqld_safe -user = mysql &
```

Si s'instal·la des de binaris (RPM, Pkg , etc.), s'ha de fer el següent:

```
cd DirectorioInstalacionMysql
./scripts/mysql_install_db
./bin/mysqld_safe user = mysql &
```

L'*script* `mysql_install_db` crea la base de dades `mysql` i `mysqld_safe` arrenca el servidor `mysqld`. A continuació, es poden provar totes les ordres donades anteriorment per Debian, excepte la primera, que és la que arrenca el servidor. A més, si s'han instal·lat els *tests*, es podran executar amb `cd sql-bench` i després, amb `run-all-tests`. Els resultats es trobaran en el directori `sql-bech/Results` per a comparar-los amb altres DB.

1.3.3. El programa monitor (client) mysql

El client `mysql` es pot utilitzar per a crear i utilitzar DB simples, és interactiu i permet connectar-se al servidor, executar cerques i visualitzar els resultats. També funciona en mode *batch* (com un *script*), on les ordres se li passen a través d'un arxiu. Per a veure totes les opcions de l'ordre, es pot executar `mysql -help`. Podrem portar a terme una connexió (local o remota) amb l'ordre `mysql`, per exemple, per a una connexió per la interfície de xarxa, però des de la mateixa màquina:

```
mysql -h localhost -u usuari -p [NomDB]
```

Si no es posa l'últim paràmetre, no se selecciona cap DB. Una vegada dintre, el `mysql` posarà un *prompt* (`mysql>`) i esperarà que introduïm alguna ordre (pròpia i SQL), per exemple, *help*. A continuació, donarem una sèrie d'ordres per a provar el servidor (recordeu posar sempre el `'` per a acabar l'ordre):

```
mysql> SELECT VERSION(), CURRENT_DATE;
Es poden fer servir majúscules o minúscules.
mysql> SELECT SIN(PI()/4), (4+1)*5;
Calculadora.
mysql> SELECT VERSION(); SELECT NOW();
Múltiples ordres en la mateixa línia
mysql> SELECT
-> USER()
-> ,
-> CURRENT_DATE;
o en múltiples línies.
mysql> SHOW DATABASES;
Mostra les DB disponibles.
mysql> USE test
Canvia la DB.
mysql> CREATE DATABASE nteum; USE nteum;
Crea i selecciona una DB denominada nteum.
mysql> CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20),
-> species VARCHAR(20), sex CHAR(1), birth DATE, death DATE);
Crea una taula dintre de nteum.
mysql> SHOW TABLES;
Mostra les taules.
mysql> DESCRIBE pet;
Mostra la definició de la taula.
mysql> LOAD DATA LOCAL INFILE "pet.txt" INTO TABLE pet;
Carrega dades des de pet.txt en pet. L'arxiu pet.txt ha de tenir un registre per línia
separat per tabulacions de les dades, d'acord amb la definició de la taula (data en format
AAAA-MM-DD).
mysql> INSERT INTO pet
-> VALUES ('Marcià', 'Estela', 'gat', 'f', '1999-03-30', NULL);
Carrega les dades in-line.
mysql> SELECT * FROM pet; Mostra les dades de la taula.
mysql> UPDATE pet SET birth = "1989-08-31" WHERE name = "Browser";
Modifica les dades de la taula.
mysql> SELECT * FROM pet WHERE name = "Browser";
Mostra selectiva.
mysql> SELECT name, birth FROM pet ORDER BY birth;
Mostra ordenada.
mysql> SELECT name, birth FROM pet WHERE MONTH(birth) = 5;
Mostra selectiva amb funcions.
mysql> GRANT ALL PRIVILEGES ON *.* TO marcia@localhost -> IDENTIFIED
BY 'passwd' WITH GRANT OPTION; Crea un usuari marcia en la DB. Ho ha de fer el root
de la DB. També es pot fer directament amb:
mysql> INSERT INTO user (Host, User, Password) ->
VALUES ('localhost', 'marcià', 'passwd');
mysql> select user, host, password from mysql.user;
Mostra els usuaris, host de connexió i passwd (també es poden posar les ordres en minús-
cules).
mysql> delete from mysql.user where user=";
Esborra els usuaris anònims (sense nom en la informació anterior).
```

1.3.4. Cas d'ús: processament de dades amb MySQL

Considerarem les dades del banc mundial de dades i concretament les d'Internet: amplada de banda internacional (bits per persona). Des de l'adreça web <http://datos.bancomundial.org/indicador> podem descarregar un full de dades

que tindrà una informació com (la informació mostra el parcial d'un total de 196 països i des de l'any 1960):

```
País ... 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010
...
Espanya 297,14 623,61 1126,83 1938,19 2821,65 2775,71 6058,60 11008,05...
```

Es crearà una base de dades en MySQL per a importar aquestes dades i generar les llistes que calculin la mitjana per país i la mitjana anual, i ordenin de major a menor el volum per a l'any 2008; i una llista dels 10 països amb major utilització d'Internet per any. Sobre aquestes últimes dades, s'ha de fer una classificació dels cinc països amb major utilització d'Internet des que existeixen dades (es mostren les ordres més rellevants i no per a totes les dades).

```
mysql -u root -p
Welcome to the MySQL monitor. Commands end with ; or \g.
Server version: 5.5.35-0+wheezy1 (Debian)
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.
mysql> create database bancomundial;

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bancomundial |
| mysql |
+-----+
3 rows in set (0.00 sec)

mysql> use bancomundial;
Database changed
mysql> create table paisbai(pais varchar(100), 1960
varchar(100), 1961 varchar(100), 1962 varchar(100), 1963
varchar(100), 1964 varchar(100), 1965 varchar(100), 1966
varchar(100), 1967 varchar(100), 1968 varchar(100), 1969
varchar(100), 1970 varchar(100), 1971 varchar(100), 1972
varchar(100), 1973 varchar(100), 1974 varchar(100), 1975
varchar(100), 1976 varchar(100), 1977 varchar(100), 1978
varchar(100), 1979 varchar(100), 1980 varchar(100), 1981
varchar(100), 1982 varchar(100), 1983 varchar(100), 1984
varchar(100), 1985 varchar(100), 1986 varchar(100), 1987
varchar(100), 1988 varchar(100), 1989 varchar(100), 1990
varchar(100), 1991 varchar(100), 1992 varchar(100), 1993
varchar(100), 1994 varchar(100), 1995 varchar(100), 1996
varchar(100), 1997 varchar(100), 1998 varchar(100), 1999
varchar(100), 2000 varchar(100), 2001 varchar(100), 2002
varchar(100), 2003 varchar(100), 2004 varchar(100), 2005
varchar(100), 2006 varchar(100), 2007 varchar(100), 2008
varchar(100), 2009 varchar(100), 2010 varchar(100) );

mysql>
CREATE TABLE 'bancomundial'. 'paisbai' ('pais' VARCHAR(100) NOT
NULL, '1960' VARCHAR(100) NOT NULL, '1961' VARCHAR(100) NOT
NULL, '1962' VARCHAR(100) NOT NULL, '1963' VARCHAR(100) NOT
NULL, '1964' VARCHAR(100) NOT NULL, '1965' VARCHAR(100) NOT
NULL, '1966' VARCHAR(100) NOT NULL, '1967' VARCHAR(100) NOT
NULL, '1968' VARCHAR(100) NOT NULL, '1969' VARCHAR(100) NOT
NULL, '1970' VARCHAR(100) NOT NULL, '1971' VARCHAR(100) NOT
NULL, '1972' VARCHAR(100) NOT NULL, '1973' VARCHAR(100) NOT
NULL, '1974' VARCHAR(100) NOT NULL, '1975' VARCHAR(100) NOT
NULL, '1976' VARCHAR(100) NOT NULL, '1977' VARCHAR(100) NOT
NULL, '1978' VARCHAR(100) NOT NULL, '1979' VARCHAR(100) NOT
NULL, '1980' VARCHAR(100) NOT NULL, '1981' VARCHAR(100) NOT
```

```

NULL, '1982' VARCHAR(100) NOT NULL, '1983' VARCHAR(100) NOT
NULL, '1984' VARCHAR(100) NOT NULL, '1985' VARCHAR(100) NOT
NULL, '1986' VARCHAR(100) NOT NULL, '1987' VARCHAR(100) NOT
NULL, '1988' VARCHAR(100) NOT NULL, '1989' VARCHAR(100) NOT
NULL, '1990' VARCHAR(100) NOT NULL, '['...]
```

```
mysql> descriu paisbai;
```

```

+-----+
|Field | Type          |Null |K| Def. |E|
| pais | varchar(100)  | YES | | NULL | |
| 1960 | varchar(100)  | YES | | NULL | |
| 1961 | varchar(100)  | YES | | NULL | |
| 1962 | varchar(100)  | YES | | NULL | |
| 1963 | varchar(100)  | YES | | NULL | |
| 1964 | varchar(100)  | YES | | NULL | |
| 1965 | varchar(100)  | YES | | NULL | |
...
| 2009 | varchar(100)  | YES | | NULL | |
| 2010 | varchar(100)  | YES | | NULL | |
+-----+
```

1. Per a carregar les dades, es pot entrar en <http://localhost/phpmyadmin> i importar les dades des d'un fitxer a la base de dades bancomundial o des de la línia del mysql.

```
LOAD DATA LOCAL INFILE '/tmp/php05Dhpl' REPLACE INTO TABLE 'paisbai'
```

```
FIELDS TERMINATED BY ';' ;
```

```
ENCLOSED BY '"' ;
```

```
ESCAPED BY '\\'
```

```
LINES TERMINATED BY '\n';
```

2. Llista que permet calcular la mitjana per país.

```

consulta SQL: SELECT 'pais',
('1960'+ '1961'+ '1962'+ '1963'+ '1964'+ '1965'+ '1966'+ '1967'+ '1968'+
'1969'+ '1970'+ '1971'+ '1972'+ '1973'+ '1974'+ '1975'+ '1976'+ '1977'+
'1978'+ '1979'+ '1980'+ '1981'+ '1982'+ '1983'+ '1984'+ '1985'+ '1986'+ '1
987'+ '1988'+ '1989'+ '1990'+ '1991'+ '1992'+ '1993'+ '1994'+ '1995'+ '19
96'+ '1997'+ '1998'+ '1999'+ '2000'+ '2001'+ '2002'+ '2003'+ '2004'+ '200
5'+ '2006'+ '2007'+ '2008'+ '2009'+ '2010')/51 as mitjanapais FROM
'paisbai' LIMIT 0, 30 ;
```

```
Fileres: 30
```

```

país          mitjanapais
Afganistan    0.0203921568627
Albània        7.3696078431373
Algèria        0.4425490196078
```

3. Generar una llista que visualitzi la mitjana anual.

```

consulta SQL: SELECT AVG ('1989') AS '1989', AVG('1990') as
'1990', AVG('1991') as '1991', AVG('1992') as '1992',
AVG('1993') as '1993', AVG('1994') as '1994', AVG('1995') as
'1995', AVG('1996') as '1996', AVG('1997') as '1997',
AVG('1998') as '1998', AVG('1999') as '1999', AVG('2000') as
'2000', AVG('2001') as '2001', AVG('2002') as '2002',
AVG('2003') as '2003', AVG('2004') as '2004', AVG('2005') as
'2005', AVG('2006') as '2006', AVG('2007') as '2007',AVG('2008')
as '2008', AVG('2009') as '2009', AVG('2010') as '2010' FROM
'paisbai';
Fileres: 1
1989 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000
0.0001 0.000 0.000 0.001 0.0019 0.005 0.044 0.158 0.6547 1.3870 27.483 126.9760 387.70
```

3. Generar una llista que visualitzi l'ordre de major a menor volum per a l'any 2008.

```
SELECT 'pais', '2008' FROM 'paisbai' ORDER BY '2008' DESC LIMIT 0 , 30;
```

```

+-----+
| pais      | 2008      |
| Lituània  | 9751.01   |
| Mongòlia  | 946.53    |
| Romania   | 9110.51   |
| Zimbabwe  | 9.71      |
...
| Butan     | 65.52     |
| Hongria   | 5977.17   |
| Vietnam   | 580.72    |
+-----+
```


4. Generar una llista dels 10 països amb major utilització d'Internet per any.

Alternativa 1.

```
SELECT 'país' , SUM( '1989' ) AS '1989' , SUM( '1990' ) AS
'1990' , SUM( '1991' ) AS '1991' , SUM( '1992' ) AS '1992' ,
SUM( '1993' ) AS '1993' , SUM( '1994' ) AS '1994' ,
SUM( '1995' ) AS '1995' , SUM( '1996' ) AS '1996' ,
SUM( '1997' ) AS '1997' , SUM( '1998' ) AS '1998' ,
SUM( '1999' ) AS '1999' , SUM( '2000' ) AS '2000' ,
SUM( '2001' ) AS '2001' , SUM( '2002' ) AS '2002' ,
SUM( '2003' ) AS '2003' , SUM( '2004' ) AS '2004' ,
SUM( '2005' ) AS '2005' , SUM( '2006' ) AS '2006' ,
SUM( '2007' ) AS '2007' , SUM( '2008' ) AS '2008' ,
SUM( '2009' ) AS '2009' , SUM( '2010' ) AS '2010'
FROM 'paisbai'
ORDER BY 'pais' DESC
LIMIT 0 , 10;
```

Alternativa 2:

```
SELECT 'pais' , MAX( '1989' ) AS '1989' , MAX( '1990' ) AS
'1990' , MAX( '1991' ) AS '1991' , MAX( '1992' ) AS '1992' ,
MAX( '1993' ) AS '1993' , MAX( '1994' ) AS '1994' ,
MAX( '1995' ) AS '1995' , MAX( '1996' ) AS '1996' ,
MAX( '1997' ) AS '1997' , MAX( '1998' ) AS '1998' ,
MAX( '1999' ) AS '1999' , MAX( '2000' ) AS '2000' ,
MAX( '2001' ) AS '2001' , MAX( '2002' ) AS '2002' ,
MAX( '2003' ) AS '2003' , MAX( '2004' ) AS '2004' ,
MAX( '2005' ) AS '2005' , MAX( '2006' ) AS '2006' ,
MAX( '2007' ) AS '2007' , MAX( '2008' ) AS '2008' ,
MAX( '2009' ) AS '2009' , MAX( '2010' ) AS '2010'
FROM 'paisbai'
GROUP BY 'pais'
LIMIT 0 , 10;
```

país	promig
Luxemburg	284474.679628
Hong Kong	21468.6420499333
San Marino	5464.22342423529
Països Baixos	3949.18559792941
Dinamarca	3743.7899214
Estònia	3118.98744675686
Suècia	2967.78367829608
Regne Unit	1902.25120777059
Suïssa	1897.13803142745
Bèlgica	1781.95881669216

1.3.5. Administració de MySQL

MySQL disposa d'un arxiu de configuració en `/etc/mysql/my.cnf` (en Debian), al qual es poden canviar les opcions per defecte de la DB, com per exemple, el port de connexió, l'usuari, la contrasenya dels usuaris remots, els arxius de registre (*logs*), els arxius de dades, si accepta connexions externes, etc. Pel que fa a la seguretat, s'han de prendre algunes precaucions:

- 1) No donar a ningú (excepte a l'usuari *root* de MySQL) accés a la taula *user* dins de la DB MySQL, ja que aquí es troben les contrasenyes dels usuaris que podrien utilitzar-se amb altres finalitats.
- 2) Verificar `mysql -o root`. Si s'hi pot accedir, significa que l'usuari *root* no té contrasenya. Per a canviar-ho, es pot fer:

```
mysql -u root mysql
mysql> UPDATE user SET Password = PASSWORD('new_password')
-> WHERE user = 'root';
mysql> FLUSH PRIVILEGES;
```

Ara, per a connectar-se com a root: `mysql -o root -p mysql`

3) Comprovar la documentació* respecte a les condicions de seguretat i de l'entorn de xarxa per a evitar problemes d'atacs o intrusions.

*http://dev.mysql.com/usingmysql/get_started.html

4) Per a fer còpies de la base de dades, es pot utilitzar l'ordre:

```
mysqldump --tab = /DirectoriDestinació --opt NomDB
o també:
mysqlhotcopy NomDB /DirectoriDestinació
```

Així mateix, és possible copiar els arxius amb extensió FRM, MYD i MYI amb el servidor parat. Per a recuperar, es pot executar mitjançant l'ordre `REPAIR TABLE` o `myisamchk -r`, la qual cosa funcionarà la majoria de vegades. En cas contrari, es podrien copiar els arxius guardats i arrencar el servidor. Hi ha altres mètodes alternatius en funció d'allò que es vulgui recuperar, com la possibilitat de guardar/recuperar part de la DB (consulteu la documentació).

1.3.6. Interfícies gràfiques

Per a MySQL hi ha gran quantitat d'interfícies gràfiques, fins i tot pròpies del paquet MySQL. Una d'aquestes és **MySQL Workbench***, que és una aplicació potent per al disseny, administració i control de bases de dades basades en MySQL (inclou el *Database Administration* que reemplaça l'anterior MySQL Administrator). Aquesta aplicació és un conjunt d'eines per als desenvolupadors i administradors de BD que integra el desenvolupament, la gestió, el control i el manteniment de manera simple i en un mateix entorn de la BD. Les parts principals són: disseny i modelatge de BD, desenvolupament en SQL (reemplaça el MySQL Query Browser), administració de la BD (reemplaça MySQL Administrator) i migració de BD.

Enllaç d'interès

Es pot trobar tota la documentació per a la instal·lació i posada en marxa de Mysql Administrator en <http://dev.mysql.com/downloads/workbench/>.

*<http://www.mysql.com/downloads/workbench>

Per a usuaris recentment iniciats (o experts) recomanem, per la seva facilitat en l'ús i simplicitat, **PhpMyAdmin** (inclosa en la majoria de distribucions, també Debian), que és una eina de programari lliure escrit en PHP per a gestionar l'administració de MySQL mitjançant el web. PhpMyAdmin és compatible amb un ampli conjunt d'operacions en MySQL com, per exemple, gestió de bases de dades, taules, camps, relacions, índexs, usuaris, permisos, etc., i també té la capacitat d'executar directament qualsevol sentència SQL. La seva instal·lació es pot fer des de la gestió de programes de distribució de treball (`apt/rpm`, `yum/aptitude`, etc.) que durant el procés d'instal·lació demanarà amb quins servidors de web es vol fer la integració i l'usuari de la base de dades. Una vegada instal·lada, es pot iniciar mitjançant

<http://localhost/phpmyadmin>, que demanarà l'usuari i la contrasenya del servidor (indicats en els passos anteriors).

Hi ha altres eines que permeten fer tasques similars, com la que ja hem comentat en la secció de PostgreSQL com **Webmin**, que permet gestionar i administrar bases de dades MySQL (incloent el mòdul corresponent). Si bé aquest paquet ja no s'inclou amb algunes distribucions (p. ex., Debian/Ubuntu), es pot descarregar des de <http://www.webmin.com>. Durant la instal·lació, el Webmin avisarà que l'usuari principal serà el *root* i utilitzarà la mateixa contrasenya que el *root* del sistema operatiu. Serà possible connectar-se, per exemple, des d'un navegador <https://localhost:10000>, el qual sol·licitarà acceptar (o denegar) la utilització del certificat per a la comunicació SSL i, a continuació, mostrarà tots els serveis que pot administrar, entre els mateixos Mysql Data Base Server.

1.4. MariaDB

MariaDB[9] és un sistema de gestió de bases de dades derivat de MySQL amb llicència GPL que incorpora dos mecanismes d'emmagatzematge nous: Aria (que reemplaça amb amplis avantatges a MyISAM) i XtraDB (que substitueix InnoDB, l'actual del Mysql). Com a *fork* de MySQL manté una alta compatibilitat ja que posseeix les mateixes ordres, interfícies, API i biblioteques amb l'objectiu que es pugui canviar un servidor per un altre. La motivació dels desenvolupadors de fer un *fork* de MySQL va ser la compra de Sun Microsystems per part d'Oracle el 2010 (Sun prèviament havia comprat MySQL AB que és la companyia que desenvolupa i manté MySQL) amb l'objectiu de mantenir aquesta sobre GPL, ja que estaven/estan convençuts que l'únic interès d'Oracle en MySQL era reduir la competència que MySQL donava al seu producte comercial (Oracle DB).

MariaDB és equivalent a les mateixes versions de MySQL (MySQL 5.5 = MariaDB 5.5, no obstant això, la versió actual és MariaDB10.1) i té una sèrie d'avantatges en relació amb MySQL que resumim a continuació*:

*<https://mariadb.com/kb/en/mariadb/mariadb-vs-mysql-features/>

1) Mecanismes d'emmagatzematge: a més dels estàndard (MyISAM, Blackhole, CSV, Memory i Archive) s'inclouen Aria (alternativa a MyISAM resistent a caigudes), XtraDB (reemplaçament directe d'InnoDB), FederatedX (reemplaçament directe de Federated), OQGRAPH, SphinxSE, Cassandra (NoSQL i en MariaDB 10.0) TokuDB (des de MariaDB 5.5), i s'estan treballant en altres NoSQL, i les últimes CONNECT, SEQUENCE i Spider (només a partir de MariaDB 10.0).

2) Facilitat d'ús i velocitat: inclou noves taules i opcions per a generar estadístiques d'índexs i taules, processos, gestió del temps en microsegons, característiques NoSQL (p. ex., HandlerSocket), columnes dinàmiques i subqueries. En relació amb la velocitat de processament, es poden veure les comparacions en relació amb MySQL en <https://mariadb.com/blog/mariadb-53-optimizer-benchmark>.

3) Tests, errors i alertes: inclou un extens conjunt de tests en la distribució que permeten analitzar diferents combinacions de configuració i sistema operatiu. Aquests tests han permès reduir notablement els errors i alertes per la qual cosa es tradueix en una BD altament estable i segura en relació amb l'execució i funcionament.

4) Llicència: el codi en MariaDB està sota GPL, LPGL o BSD i no disposa de mòduls tancats com MySQL *Enterprise Ed* (de fet, tot el codi tancat en MySQL 5.5 E.Ed. està en MariaDB com a *open source version*). MariaDB inclou tests per a tots els errors solucionats, mentre que Oracle no fa el mateix amb MySQL 5.5 i tots els errors i plans de desenvolupaments són públics i els desenvolupadors pertanyen a la comunitat.

És important tenir en compte que MariaDB és desenvolupada per la comunitat, i la responsabilitat de desenvolupament i manteniment són a càrrec de SkySQL Corporation Ab, empresa fundada el 2010 per les mateixes persones de MySQL AB (David Axmark, Michael 'Monty' Widenius i Kaj Arnö). Aquesta empresa es va fusionar amb el Monty Program el 2013 (programa fundat per Monty Widenius -desenvolupador de MySQL juntament amb D. Axmark- després que va vendre la companyia a Sun MS i va decidir escriure MariaDB). El 2014, SkySQL crea una altra marca anomenada MariaDB AB i ofereix MariaDB *Enterprise*, MariaDB *Enterprise Cluster* i *MaxScale* sota un model diferent de negoci que la llicència de programari com a MySQL, per a dirigir-se a un mercat empresarial garantint fiabilitat i confiabilitat en aplicacions de missió crítica i alt rendiment. <https://mariadb.com/about>

1.4.1. Instal·lació MariaDB sobre Debian

La instal·lació és summament senzilla i tal com s'indica en [10] (els paquets estaran en el repositori Debian a partir de la propera versió Jessie). Se selecciona la distribució, versió del SO i repositori, i s'executa:

```
Si tenim BD en MySQL, fer una còpia de seguretat parant els serveis primer i
després abocant la BD:
service apache2 stop; service mysql stop
mysqldump -u root -p --all-databases > mysqlbackup.sql
Configurem el repositori:
apt-get install python-software-properties
apt-key adv --recv-keys --keyserver keyserver.ubuntu.com 0xc9cb082a1bb943db
add-apt-repository 'deb http://mirror.vpsfree.cz/mariadb/repo/10.1/debian wheezy main'
Després d'importar la clau, actualitzem el repositori i instal·lem:
apt-get update
apt-get install mariadb-server
service apache2 start
Podrem mirar la versió amb: mysql --version
mysql Ver 15.1 Distrib 10.1.0-MariaDB, for debian-linux-gnu (x86_64) using readline 5.1
O les bases de dades existents amb:
mysql -u root -p -Be 'show databases'
I si accedim a phpmyadmin veurem en "Home"
MySQL
Server: Localhost via UNIX socket
Server version: 10.1.0-MariaDB-1~wheezy
Protocol version: 10
```

```
User: root@localhost
MySQL charset: UTF-8 Unicode (utf8)
```

Veurem que si tenim instal·lat `mysql-server`, el desinstal·la i el reemplaça per `mariadb`, fent les mateixes preguntes de les bases de dades creades (usuari i *passwd*).

1.5. SQLite

Com ja vam dir en la introducció, SQLite és un sistema de gestió de bases de dades relacional compatible amb ACID, continguda en una biblioteca (escrita en C) sota llicència *Public Domain* (<http://www.sqlite.org/copyright.html>). Com a gran diferència dels sistemes de gestió de bases de dades client-servidor, el motor de SQLite no és un procés independent del programa que desitja portar a terme un accés a la base de dades, sinó una biblioteca que s'enllaça juntament amb el programa i passa a ser-ne part com qualsevol altra biblioteca (p. ex., igual que la `/lib/x86_64-linux-gnu/libc.so.6` en Debian per a un programa en C). El programa que necessita la funcionalitat de SQLite simplement fa la crida a subrutines i funcions tenint com a avantatge substancial la reducció de la latència d'accés a la base de dades ja que les crides a funcions són més eficients que la comunicació entre processos. El conjunt de la base de dades (definicions, taules, índexs i les pròpies dades) és guardat en un arxiu fitxer sobre la màquina *host* i la coherència en les transaccions s'obté bloquejant tot el fitxer al principi de cada transacció. La biblioteca implementa la major part de l'estàndard SQL-92, manté les característiques necessàries perquè una sèrie d'instruccions puguin ser considerades com una transacció (*ACID compliant*) i el seu disseny permet que diversos processos o *threads* puguin accedir a la mateixa base de dades sense problemes on els accessos de lectura poden ser servits en paral·lel i un accés d'escriptura només pot ser servit si no s'està fent cap altre accés de manera concurrent (o donarà un error o es podrà reintentar durant un cert *timeout* programable).[11]

En Debian, s'inclou un paquet que implementa una interfície en línia d'ordres per a SQLite2/3 anomenat `sqlite` (versió2) o `sqlite3` (`apt-get install sqlite3`) per a accedir i modificar BD SQLite a més d'un conjunt de biblioteques que permeten la interacció entre el llenguatge i l'aplicació, per exemple, `python-sqlite`, `ruby-sqlite`, `php5-sqlite` com a interfícies per a SQLite des de `python`, `ruby` i `php`, respectivament. A més, hi ha altres eines per a dissenyar, crear o editar una base de dades compatible amb SQLite, com per exemple SQLite Database Browser (GPL) (`apt-get install sqlitebrowser`), que permet de manera simple i visual accedir als fitxers que conté la base de dades [12]. Entre els principals usuaris de SQLite, trobem* Adobe (Photoshop Lightroom, AIR), Airbus (*flight software* A350), Apple (OSX, iPhone, iPod), Dropbox, Firefox, Thunderbird, Google (Android, Chrome), PHP, Python, Skype i Tcl/Tk, entre d'altres.

*<http://www.sqlite.org/famous.html>

Per a instal·lar la biblioteca SQLite en Debian, fem

```
apt-get install libsqlite0-dev
```

(que inclou la biblioteca i els arxius *.h* per desenvolupar aplicacions en C, si només necessitem la biblioteca amb *libsqlite0* n'hi ha prou). Per a visualitzar per exemple una de les bases de dades de Firefox fem:

```
Busquem el nostre profile de firefox/iceweasel en l'usuari de login: cd .mozilla/firefox/*.default
Obrim, per exemple, la base de dades:: sqlite3 places.sqlite
SQLite version 3.7.13 2012-06-11 02:05:22
Enter ".help" for instructions. Enter SQL statements terminated with a ";"
Mirem la taules amb .tables
moz_anno_attributes  moz_favicons          moz_items_annos
moz_annos            moz_historyvisits      moz_keywords
moz_bookmarks        moz_hosts              moz_places
moz_bookmarks_roots  moz_inpuhistory
Mirem la taula moz_hosts:  SELECT * FROM moz_hosts;
1|mozilla.org|140|0|
2|sysdw.nteum.org|11180|1|
3|127.0.0.1|2000|1|
4|10.0.0.58|0|0|
Mirem la taula moz_bookmarks (abans fem un bookmark, per exemple a SQLite.org:
SELECT * FROM moz_bookmarks;
...
24|1|164|2|4|SQLite Home Page||1404403846763242|1404403846777354|yfpoHwYrL7Ys
```

En l'adreça web <http://www.sqlite.org/quickstart.html> tindrem una guia ràpida de com accedir mitjançant les diferents interfícies de programació a la base de dades i la documentació per a treballar amb la línia d'ordres en l'adreça web <http://www.sqlite.org/cli.html>. La documentació la podem trobar en <http://www.sqlite.org/docs.html>.

1.6. Source Code Control System

Els sistemes de control de revisions (o també coneguts com a *control de versions/codi font*) es refereixen a la gestió dels canvis en els arxius, programes, pàgines o qualsevol informació que és modificada i actualitzada, ja sigui per un únic usuari per a portar un control del que ha fet i quan o per múltiples usuaris que treballen de manera simultània en el mateix projecte. Els canvis s'identifiquen generalment per un nombre o codi, denominat *nombre de revisió*, *nivell de revisió*, *revisió* i el sistema de control permet tornar enrere, acceptar un canvi, publicar-lo o saber qui l'ha fet i quin ha estat. El sistema treballa amb marques de temps, repositoris (per a guardar les diferents versions) i usuaris i permetrà, en funció d'aquests paràmetres, comparar els repositoris/arxius, restaurar-los i/o fusionar-los. En l'actualitat, és una tècnica aplicada en grans aplicacions informàtiques (com, per exemple, en Google Docs o Wikipedia -o en qualsevol wiki-), però hi ha diferents aplicacions que poden treballar com a servidors o aplicacions internes o en mode client servidor en funció de les necessitats del projecte/usuaris com per exemple CVS (uns dels inicials juntament amb RCS), Subversion (molt estès), Git (molt utilitzat en projectes *Open*

Source) o Mercurial (molt utilitzat pel seu aspecte de distribuït –no és necessari un servidor).

1.6.1. Concurrent Versions System (CVS)

El *Concurrent Versions System* (CVS) és un sistema de control de versions que permet mantenir versions antigues d'arxius (generalment codi font), guardant un registre (*log*) de qui, quan i per què van ser efectuats els canvis.

A diferència d'altres sistemes, CVS no treballa amb un arxiu/directori per vegada, sinó que actua sobre col·leccions jeràrquiques dels directoris que controla. El CVS té per objectiu ajudar a gestionar versions de programari i controla l'edició concurrent d'arxius font per múltiples autors. El CVS utilitza internament un altre paquet anomenat *RCS* (*Revision Control System*) com una capa de baix nivell. Si bé el RCS pot ser utilitzat de manera independent, això no s'aconsella, ja que CVS, a més de la pròpia funcionalitat, presenta totes les prestacions de RCS, però amb notables millores quant a l'estabilitat, el funcionament i el manteniment. Entre aquestes, cal destacar: funcionament descentralitzat (cada usuari pot tenir el seu propi arbre de codi), edició concurrent, comportament adaptable mitjançant *shell scripts*, etc. [13].

En primer lloc, s'ha d'instal·lar el Concurrent Versions System (CVS) des de la distribució i hauréu d'instal·lar també OpenSSH, si es vol utilitzar conjuntament amb CVS per a accés remot. Les variables d'entorn `EDITOR` `CVSROOT` han d'estar inicialitzades, per exemple, en `/etc/profile` (o en `.bashrc` o `.profile`):

```
export EDITOR = /bin/vi
export CVSROOT = /usr/local/cvsroot
```

Òbviament, els usuaris poden modificar aquestes definicions utilitzant `~/.bashrc` i s'ha de crear el directori on hi haurà el repositori i configurar els permisos; com *root*, cal fer, per exemple:

```
export CVSROOT = /usr/local/cvsroot
groupadd cvs
useradd -g cvs -d $CVSROOT cvs
mkdir $CVSROOT
chgrp -R cvs $CVSROOT
chmod o-rwx $CVSROOT
chmod ug+rwx $CVSROOT
```

Per a inicialitzar el repositori i posar-hi arxiu de codi:

```
cvs -d /usr/local/cvsroot init
```

`cvs init` tindrà en compte no sobre escriure mai un repositori ja creat per a evitar pèrdues d'altres repositoris. Després, s'hauran d'afegir els usuaris que treballaran amb el

CVS al grup cvs; per exemple, per a agregar l'usuari nteum:

```
usermod -G cvs,nteum
```

Ara l'usuari nteum haurà d'introduir els seus arxius en el directori del repositori (en el nostre cas /usr/local/cvsroot):

```
export EDITOR = /bin/vi
export CVSROOT = /usr/local/cvsroot
export CVSREAD = yes
cd directori_d'originals
cvs import NomDelRepositori vendor_1_0 rev_1_0
```

El nom del repositori pot ser un identificador únic o també hi ha la possibilitat que sigui usuari/projecte/xxxx si és que l'usuari desitja tenir organitzats els seus repositoris.

Això crearà un arbre de directoris en CVSROOT amb aquesta estructura i afegeix un directori (/usr/local/cvsroot/NomDelRepositori) en el repositori amb els arxius que a partir d'aquest moment estaran en el repositori. Una prova per a saber si s'ha emmagatzemat tot correctament és emmagatzemar una còpia en el repositori, crear després una còpia des d'allà i comprovar les diferències. Per exemple, si els originals estan en el directori_de l'usuari/dir_org i es desitja crear un repositori com primer_cvs/proj, s'hauran d'executar les següents ordres:

```
cd dir_org    Canviar al directori del codi font original.
cvs import -m "Fonts originals" primer_cvs/proj usuariX vers0
Crea el repositori en primer_cvs/proj amb usuariX i vers0.
cd..    Canviar al directori superior de dir_org.
cvs checkout primer_cvs/proj
Generar una còpia del repositori. La variable CVSROOT ha d'estar inicialitzada, en cas
contrari, s'haurà d'indicar tot el camí.
diff -r dir_org primer_cvs/proj
Mostra les diferències entre l'un i l'altre. No n'hi haurà d'haver cap excepte pel directori
primer_cvs/proj/CVS que ha creat el CVS.
rm -r dir_org
Esborra els originals (feu sempre una còpia de seguretat per motius de seguretat i per
tenir una referència d'on es va iniciar el treball amb el CVS).
```

Estructura dels directoris, arxius i branques:

```
/home/nteum/primer_cvs/proj: a.c b.c c.c Makefile
    usuariX, vers0.x -> Treballar només amb aquest directori després de l'import
/home/nteum/dir_org/: a.c b.c c.c Makefile                Després del checkout, s'haurà d'esborrar

/usr/local/cvsroot/primer_cvs/proj/: a.c b.c c.c Makefile usuariX, vers0.1
/usr/local/cvsroot/primer_cvs/proj/: a.c b.c c.c Makefile usuariX, vers0.1.1 -> Branch 1.1

/usr/local/cvsroot/primer_cvs/proj/: a.c b.c c.c Makefile usuariX, vers0.2
/usr/local/cvsroot/primer_cvs/proj/: a.c b.c c.c Makefile usuariX, vers0.x
```

El fet d'esborrar els originals no sempre és una bona idea, excepte en aquest cas, després que s'hagi verificat que estan en el repositori, perquè no s'hi treballi per distracció i perquè els canvis no quedin reflectits sobre el CVS. Sobre màquines on els usuaris volen accedir (per ssh) a un servidor CVS remot, s'haurà de fer:


```
export CVSROOT = ":ext:user@CVS.server.com:/home/cvsroot";  
export CVS_RSH = "ssh"
```

on `user` és el *login* de l'usuari i `cvs.server.com`, el nom del servidor en el qual està CVS. CVS ofereix una sèrie d'ordres (s'obtenen amb `cvs cmd` opcions...) per a treballar amb el sistema de revisions, entre aquestes: `add`, `checkout`, `update`, `remove`, `commit` i `diff`.

Ordres de CVS

`cvs checkout` és l'ordre inicial i crea la seva còpia privada del codi font per després treballar-hi sense interferir en el treball d'altres usuaris (com a mínim es crea un subdirectori on estaran els arxius).

`cvs update` s'ha d'executar de l'arbre privat quan cal actualitzar les seves còpies d'arxius font amb els canvis que altres programadors han fet sobre els arxius del repositori.

`cvs add file` és una ordre necessària quan cal agregar nous arxius en el seu directori de treball sobre un mòdul on ja s'ha fet prèviament un *checkout*. Aquests arxius s'enviaran al repositori CVS quan s'executi l'ordre `cvs commit`.

`cvs import` es pot usar per a introduir arxius nous en el repositori.

`cvs remove file` s'utilitzarà per a esborrar arxius del repositori (una vegada que s'hagin esborrat de l'arxiu privat). Aquesta ordre ha d'anar acompanyada d'un `cvs commit` perquè els canvis siguin efectius, ja que es tracta de l'ordre que transforma totes les peticions dels usuaris sobre el repositori.

`cvs diff file` es pot utilitzar sense que afecti cap dels arxius implicats si es necessita verificar les diferències entre repositori i directori de treball o entre dues versions.

`cvs tag -R "versió"` es pot usar per a introduir un nombre de versió en els arxius d'un projecte i després fer un `cvs commit` i un projecte `cvs checkout -r 'version'` per a registrar una nova versió.

Una característica interessant del CVS és que pot aïllar canvis dels arxius en una línia de treball separada denominada *ramificació* o *branca* (*branch*). Quan es canvia un arxiu sobre una branca, aquests canvis no apareixen sobre els arxius principals o sobre altres branques. Més tard, aquests canvis es poden incorporar a altres branques o a l'arxiu principal (*merging*). Per a crear una nova branca, utilitzeu `cvs tag -b rel-1-0-patches` dins del directori de treball, la qual cosa assignarà a la branca el nom de `rel-1-0-patches`. La unió de branques amb el directori de treball significa utilitzar l'ordre `cvs update -j`. Consulteu les referències per a barrejar diferents branques o accedir-hi.

Exemple d'una sessió

Seguint l'exemple de la documentació donada en les referències, es mostrarà una sessió de treball (en forma general) amb CVS. Com que CVS emmagatzema tots els arxius en un repositori centralitzat, s'assumirà que ja ha estat inicialitzat anteriorment. Considerem que s'està treballant amb un conjunt d'arxius en C i un Makefile, per exemple. El compilador utilitzat és gcc i el re-

positori és inicialitzat a `gccrep`. En primer lloc, s'ha d'obtenir una còpia dels arxius del repositori a la nostra còpia privada amb:

```
cvs checkout gccrep
```

Això crearà un nou directori anomenat `gccrep` amb els arxius font. Si s'executa `cd gccrep; ls`, es veurà per exemple `CVS makefile a.c b.c c.c`, on hi ha un directori CVS que es crea per al control de la còpia privada que normalment no és necessari tocar. Després d'això, es podria utilitzar un editor per a modificar `a.c` i introduir canvis substancials en l'arxiu (consulteu la documentació sobre múltiples usuaris concurrents si es necessita treballar amb més d'un usuari en el mateix arxiu), compilar, tornar a canviar, etc. Quan es decideix que es té una versió nova amb tots els canvis introduïts en `a.c` (o en els arxius que sigui necessari), és el moment de fer una nova versió emmagatzemant `a.c` (o tots els que s'han tocat) en el repositori i fer aquesta versió disponible per a la resta d'usuaris:

```
cvs commit a.c
```

Utilitzant l'editor definit en la variable `CVSEEDITOR` (o `EDITOR` si aquesta no està inicialitzada), es podrà introduir un comentari que indiqui quins canvis s'han fet perquè serveixi d'ajuda a altres usuaris o per a recordar què és el que va caracteritzar a aquesta versió i després poder fer un històric.

Si es decideix eliminar els arxius (perquè ja es va acabar el projecte o perquè no s'hi treballarà), una forma de fer-ho és en un nivell de sistema operatiu (`rm -r gccrep`), però és millor utilitzar el propi `cvs` fora del directori de treball (nivell immediat superior): `cvs release -d gccrep`. L'ordre detectarà si hi ha algun arxiu que no ha estat enviat al repositori i, si n'hi ha i s'esborra, preguntarà si es desitja continuar o no per a evitar que es perdin tots els canvis. Per a mirar les diferències, per exemple, si s'ha modificat `b.c` i no es recorda quins canvis es van fer, es pot utilitzar `cvs diff b.c` dins del directori de treball. S'utilitzarà l'ordre del sistema operatiu `diff` per a comparar la versió `b.c` amb la versió que es té en el repositori (sempre cal recordar fer un `cvs commit b.c` si es desitja que aquestes diferències es transfereixin al repositori com una nova versió).

Múltiples usuaris

Quan més d'una persona treballa en un projecte de programari amb diferents revisions, es tracta d'una situació summament complicada perquè hi haurà ocasions en les quals més d'un usuari vulgui editar el mateix fitxer de manera simultània. Una possible solució és bloquejar el fitxer o utilitzar punts de verificació reservats (*reserved checkouts*), la qual cosa només permetrà a un usuari editar el mateix fitxer simultàniament. Per a això, s'haurà d'executar l'ordre `cvs admin -l command` (consulteu `man` per a les opcions).

CVS utilitza un model per defecte de punts no reservats (*unreserved checkouts*), que permet als usuaris editar simultàniament un fitxer del seu directori de treball. El primer d'ells que transfereixi els seus canvis al repositori ho podrà fer sense problemes, però els restants rebran un missatge d'error quan desitgin fer la mateixa tasca, per la qual cosa, hauran d'utilitzar ordres de cvs per a transferir en primer lloc els canvis al directori de treball des del repositori i després actualitzar el repositori amb els seus propis canvis. Consulteu les referències per a veure un exemple d'aplicació i altres formes de treball concurrent amb comunicació entre usuaris [13]. Com a interfícies gràfiques es poden consultar tkcvs* [34] desenvolupada en Tcl/Tk i que suporta Subversion o la també molt popular Cervisia [35] (les dues en Debian).

*<http://www.twobarleycorns.net/tkcvs.html>

1.6.2. Subversion

Com a idea inicial, **Subversion** serveix per a gestionar un conjunt d'arxius (repositori) i les seves diferents versions. És interessant remarcar que no ens importa com es guarden, sinó com s'accedeix a aquests fitxers i que és comú utilitzar una base de dades. El repositori és com un directori del qual es vol recuperar un fitxer de fa una setmana o 10 mesos a partir de l'estat de la base de dades, recuperar les últimes versions i agregar-ne una de nova. A diferència de CVS, Subversion fa les revisions globals del repositori, la qual cosa significa que un canvi en el fitxer no genera un salt de versió únicament en aquest fitxer, sinó en tot el repositori, el qual en suma un a la revisió. En Debian, haurem de fer `apt-get install subversion subversion-tools`, i si desitgem publicar els repositoris en Apache2 haurem de tenir instal·lat aquest, a més del mòdul específic `apt-get install libapache2-svn`. A més del paquet Subversion, hi ha en els repositoris (de la majoria de distribucions) paquets complementaris, com per exemple, `subversion-tools` (eines complementàries), `svn-load` (versió millorada per a la importació de repositoris), `svn-workbench` (*Workbench* per a Subversion) i `svnmailer` (eina que estén les possibilitats de les notificacions del *commit*).

Enllaç d'interès

A més del llibre disponible en <http://svnbook.red-bean.com/nightly/es/index.html>, consulteu la documentació en <http://subversion.tigris.org/servlets/ProjectDocumentList>.

Primer pas: crear el nostre repositori, usuari (considerem que l'usuari és *svuser*), grup (*svgroup*) com a *root* fer:

```
mkdir -p /usr/local/svn
adduser svuser
addgroup svgroup
chown -R root.svgroup /usr/local/svn
chmod 2775 /usr/local/svn
addgroup svuser svgroup Afegim l'usuari svuser al grup svgroup
```

Ens connectem com *svuser* i verifiquem que estem en el grup *svgroup* (amb l'ordre *id*).

```
svnadmin create /usr/local/svn/proves
```

Aquesta ordre crearà un sèrie d'arxius i directoris per a fer el control i gestió de les versions. Si no es té permís en */usr/local/svn*, es pot fer en el directori local:

```
mkdir -p $HOME/svndir
svnadmin create $HOME/svndir/proves
```

Considerant que el repositori el tenim en */usr/local/svn/proves/*, a continuació crearem un directori temporal en el nostre directori:

```
mkdir -p $HOME/svntmp/proves
```

Ens passem al directori `cd $HOME/svntmp/proves` i crearem un arxiu, per exemple:

```
echo Primer Arxiu Svn `date` > file1.txt El traslladem al repositori fent
```

dins del directori:

```
svn import file:///usr/local/svn/proves/ -m "Ver. Inicial"
```

Si l'hem creat en `$HOME/svndir/proves`, l'hauríem de reemplaçar per `file:/// $HOME/svndir/proves`. L'import copia l'arbre de directoris i el `-m` permet indicar-li el missatge de versió. Si no posem `-m`, s'obrirà un editor per a fer-ho (s'ha de posar un missatge per a evitar problemes). El subdirectori `$HOME/svntmp/proves` és una còpia del treball en repositori i és recomanable esborrar-la per a no tenir la temptació o cometre l'error de treballar amb aquesta en comptes de fer-ho amb el repositori (`rm -rf $HOME/svntmp/proves`).

Una vegada en el repositori, es pot obtenir la còpia local on podrem treballar i després pujar les còpies al repositori. Per a això fem:

```
mkdir $HOME/svn-work; cd $HOME/svn-work
svn checkout file:///home/svuser/svndir/proves
```

on veurem que tenim el directori `proves`. Es pot copiar amb un altre nom agregant al final el nom que volem. Per a afegir-hi un nou fitxer:

```
cd $HOME/svn-work/proves
echo Segon Arxiu Svn `date` > file2.txt
svn add file2.txt
svn commit -m "Nou arxiu"
```

És important remarcar que una vegada en la còpia local (`svn-work`), no s'ha d'indicar el camí (*path*). `svn add` marca per a afegir el fitxer al repositori i realment s'afegeix quan fem un `svn commit`.

Ens donarà alguns missatges que ens indicaran que és la segona versió. Si agreguem una altra línia, la `file1.txt` amb `echo `date` >> file1.txt`, després podem pujar els canvis amb `svn commit -m "Nova línia"`. És possible comparar l'arxiu local amb el del repositori. Per a fer-ho, podem agregar una tercera línia a `file1.txt` amb `echo `date` >> file1.txt` sense pujar-lo i si volem veure les diferències, podem fer `svn diff`. Aquesta ordre ens marcarà quines són les diferències entre l'arxiu local i els del repositori. Si ho carreguem amb `svn commit -m "Nova línia2"` (que generarà una altra versió), després el `svn diff` no ens donarà diferències.

També es pot utilitzar l'ordre `svn update` dins del directori per a actualitzar la còpia local. Si hi ha dos o més usuaris treballant al mateix temps i cadascun ha fet una còpia local del repositori i la modifica (fent un `commit`), quan el segon usuari vagi a fer el `commit` de la seva còpia amb les seves modificacions, li donarà un error de conflicte, ja que la còpia en el repositori és posterior a la còpia original d'aquest usuari (és a dir, hi ha hagut canvis entremig), amb la qual cosa si el segon usuari fa el `commit`, perdem les modificacions del primer. Per a això haurem de fer un `svn update`, que ens indicarà l'arxiu en conflicte i en l'arxiu en conflicte ens indicarà quin és l'arxiu i les seves parts que estan en conflicte. L'usuari haurà de decidir amb quina versió es queda i després podrà fer un `commit`. Una ordre interessant és `svn log file1.txt`, que ens mostrarà tots els canvis fets en el fitxer i les seves corresponents versions.

Un aspecte interessant és que Subversion pot funcionar conjuntament amb Apache2 (i també sobre SSL) per a accedir des d'una altra màquina o simplement mirar el repositori. La configuració d'Apache2 i SSL es va indicar en la part de servidors, però també s'explica en Debian Administration. Per

Enllaç d'interès

Consulteu els clients per a accedir a Subversion a:
<http://svnbook.xarxa-bean.com>.

a configurar-los, és necessari activar els mòduls de WebDAV (`a2enmod dav dav_fs`) i instal·lar la biblioteca com s'ha especificat abans amb `apt-get install libapache2-svn` verificant que el mòdul `dav_svn` estigui habilitat també (ho farà la instal·lació de la biblioteca). A continuació, crearem un arxiu en `/etc/apache2/conf.d/svn.conf` amb el següent contingut:

```
<location "/svn">
    DAV svn
    SVNParentPath /usr/local/svn
    SVNListParentPath On
    AuthType Basic
    AuthName "Subversion"
    AuthUserFile /etc/apache2/htpasswd
    Require valid-user
</location>
```

On hem d'apuntar cap a on es troba el nostre repositori amb validació d'usuari (ha de ser un usuari vàlid creat amb `htpasswd /etc/apache2/htpasswd user`), reiniciem Apache2 i ja tindrem aquest habilitat per a accedir mitjançant l'URL `http://sysdw.nteum.org/svn/`. Una altra forma d'accedir és a través d'un client en forma local o en forma remota, que és una opció útil per a repositoris petits i mitjans. Debian n'inclou una sèrie (alguns compatibles amb CVS) com per exemple **rapidsvn**, **subcommander**, **svnkit** (en Java), **tkcvs**, **viewvc** (aquests dos últims suporten també repositoris CVS) i **websvn** (en PHP). Si es desitja treballar en remot, s'haurà d'engegar el servidor de svn amb `svnserve -d` i canviar l'arxiu `svnserve.conf` per a permetre l'accés i el mode (p. ex., traient el comentari en la línia `anon-access = read`). Aquest arxiu es troba en el directori `/Dir_Repo/conf`, que és on hem inicialitzat el repositori (`/usr/local/svn/proves/conf/svnserve.conf` en el nostre cas).

Si desitgem treballar amb un repositori gestionat via URL, el més pràctic és crear-ne un i gestionar-lo des de la consola per a inserir els arxius i via web per a posar-los a la disposició del públic. En aquest cas, haurem de fer algunes modificacions amb els permisos perquè puguem treballar amb URL. Crearem un segon repositori per a aquest exemple, però farem que l'usuari **svuser** generi les seves còpies dins del repositori:

Com a *root*, fem:

```
mkdir /var/svn El nostre nou repositori
svnadmin create /var/svn Creem el repositori
chown -R www-data:www-data Perquè Apache pugui accedir al directori
ls -ls /var/svn
-rw-r--r-- 1 www-data www-data 229 Jul 4 20:27 README.txt
drwxr-xr-x 2 www-data www-data 4096 Jul 4 20:27 conf
drwxr-xr-x 2 www-data www-data 4096 Jul 4 20:27 dav
drwxr-xr-x 2 www-data www-data 4096 Jul 4 20:28 db
-rw-r--r-- 1 www-data www-data 2 Jul 4 20:27 format
drwxr-xr-x 2 www-data www-data 4096 Jul 4 20:27 hooks
drwxr-xr-x 2 www-data www-data 4096 Jul 4 20:27 locks
```

Per a l'autenticació, s'usa `htpasswd` (per exemple, amb la instrucció `htpasswd -c -m /etc/apatxe2/htpasswd user` on el paràmetre `-c` només s'ha de posar la primera vegada quan s'ha de crear l'arxiu). Després, crearem un arxiu en `/etc/apache2/conf.d/svn2.conf` per a alguna cosa com:

```
<location /svn2>  
  DAV svn  
  SVNPath /var/svn  
  AuthType Basic  
  AuthName "Subversion Repository"  
  AuthUserFile /etc/apache2/htpasswd  
  Require valid-user  
</location>
```

Reiniciem Apache i ja estem llestos per a importar alguns arxius, per exemple des de la consola com svuser:

```
svn import file1.txt http://sysdw.nteum.org/svn/file1.txt -m " Import  
Inicial"
```

Ens demanarà l'autenticació i ens dirà que el fitxer `file1.txt` s'ha afegit al repositori. Des de l'URL `http://sysdw.nteum.org/svn2/` veurem el `file1.txt`.

1.6.3. Git

Git és un programa de control de versions dissenyat per Linus Torvalds basat en l'eficiència i la fiabilitat del manteniment de versions d'aplicacions amb un elevat nombre arxius de codi font que, si bé es va dissenyar com un motor de baix nivell sobre el qual es poguessin escriure aplicacions d'usuari (*front ends*), s'ha convertit des de llavors en un sistema de control de versions amb funcionalitat plena a partir de la seva adopció com a eina de revisió de versions per al grup de programació del nucli Linux. Entre les característiques més rellevants trobem:

- 1) Suporta el desenvolupament no lineal i permet la gestió eficient de ramificacions i barreja de diferents versions. Git inclou eines específiques per a navegar i visualitzar un historial de desenvolupament no lineal.
- 2) Gestió distribuïda: permet que cada programador tingui una còpia local de l'historial del desenvolupament sencer i que els canvis es propaguin entre els repositoris locals.
- 3) Els repositoris poden publicar-se per HTTP, FTP, rsync o mitjançant un protocol natiu, mitjançant una connexió TCP/IP simple o a través de xifratge SSH, i permet emular servidors CVS per a interactuar amb clients CVS preexistents.
- 4) Els repositoris Subversion i svn es poden fer servir directament amb git-svn.

Instal·lació d'un repositori de Git, Gitolite i Gitweb

Instal·larem un servidor de control de versions Git sobre Debian Wheezy perquè pugui gestionar els arxius de codi font d'un equip de desenvolupament. Per a això, utilitzarem Gitolite (com a mètode de control d'accés dels nostres programadors als repositoris de programari), que és una capa d'autenticació a Git basat en ssh i que permet especificar permisos no només per repositori,

sinó també per branca o etiquetes dins de cada repositori. Finalment, instal·larem Gitweb per a l'accés als repositoris i adaptarem una interfície basada en <http://kogakure.github.io/gitweb-theme/>. [17, 18, 19]

Sobre el servidor, instal·lem els paquets: `apt-get install git-core git-doc gitolite git-daemon-run gitweb highlight`

Crearem un usuari per al servei git: `adduser git`

Adoptem la seva identitat i configurem:

```
su - git
git config --global user.name "git"
git config --global user.email git@sysdw.nteum.org
```

Es pot verificar amb `git config -l`

S'han de generar les claus públiques dels usuaris i enviar al servidor, per la qual cosa podem fer com a usuari adminp:

```
ssh-keygen
scp .ssh/id_rsa.pub git@sysdw.nteum.org:/tmp/adminp.pub
```

Si l'usuari adminp està en el mateix servidor utilitza `cp` en lloc de `scp`, però se suposa que és una altra màquina.

En el servidor, activem la clau d'adminp (i la de tots els usuaris), i com a usuari git fem

```
gl-setup /tmp/adminp.pub
```

Això obrirà un editor (d'arxiu `/home/git/.gitolite.rc`) amb indicacions en el qual haurem de canviar la línia `$REPO_UMASK = 0027`; (en comptes de 77 canviar per 27).

A continuació, canviarem els permisos:

```
chmod g+r /home/git/projects.list
chmod -R g+rx /home/git/repositories
```

Editem `/etc/group` i afegim l'usuari `www-data` al grup git modificant la línia a `git:x:1001:www-data`.

En la màquina local i com a usuari adminp (també es pot fer en la mateixa màquina del servidor en el compte de l'usuari adminp), carregarem la configuració de gitolite i agregarem dos repositoris:

```
git clone git@sysdw.nteum.org:gitolite-admin.git
cd gitolite-admin
```

Editem l'arxiu `conf/gitolite.conf` amb els repositoris:

```
repo    gitolite-admin
RW+     =    adminp

repo    testing
RW+     =    @all

repo    wiki
RW+     = @all
R       = daemon
R       = gitweb

repo    data
RW+     = adminp remix
R       = daemon
R       = gitweb
```

Després agreguem en el repositori: `git add *`

I fem el commit: `git commit -m "Repositoris wiki i data amb permís d'escriptura per a admin i remix"`

El pugem: `git push origin master`

Per a la configuració de gitweb, editem `/etc/gitweb.conf` modificant les línies indicades:

```
$projectroot = "/home/git/repositories";
$projects_list = "/home/git/projects.list";
```

Modifiquem `/etc/sv/git-daemon/run` (abans hem de fer una còpia com `run.org`, per exemple)

```
#!/bin/sh
exec 2>&1
echo 'git-daemon starting.'
exec chpst -ugitdaemon:git \
  "$(git --exec-path)"/git-daemon -verbose -reuseaddr \
  -base-path=/home/git/repositories /home/git
```

Després reiniciem el *daemon*: `sv restart git-daemon`

També reiniciem Apache2 i en l'URL `http://sysdw.nteum.org/gitweb/` tindrem els projectes.

Per a canviar l'aspecte de la pàgina web amb `http://kogakure.github.io/gitweb-theme/`. Podem descarregar l'arxiu `tgz` d'aquesta pàgina, el descomprimim i des de dins del directori copiem el contingut

```
cp * /usr/share/gitweb/static/
```

Si recarreguem el directori esborrant la *cache* (Ctrl+F5) en el navegador, hauríem de veure el nou aspecte. Un aspecte interessant és veure ara com des de l'usuari admin inserim arxius en un dels repositoris i els publiquem:

Creem un directori *hello* amb dos arxius, un arxiu anomenat *hello.c* i un altre, *library.h*

```
/* Hello.c */
#include <stdio.h>
#include "library.h"
int main (void) {printf ("Hola UOC!\n");}
/* library.h */
#ifndef DEFINITIONS_H
#define DEFINITIONS_H 1
#endif /* DEFINITIONS_H */
```

Dins del directori, executem:

```
git init
git add .
git commit -m "initial commit"
git remote add origin git@sysdw.nteum.org:wiki.git
git push origin master
```

Si actualitzem la pàgina web, veurem que ja tenim els arxius en el repositori.

Si volguéssim clonar el repositori, només hauríem de fer:

```
git clone git@sysdw.nteum.org:wiki.git
```

I on estem, crearem un directori *wiki* amb els arxius *hello.c* *library.h* (a més d'un altre *.git* que és on es troba tota la informació de Git).

1.6.4. Mercurial

Mercurial és un sistema de control de versions (escrit en la seva major part en Python i algunes parts en C -diff-) dissenyat per a facilitar la gestió d'arxius/directoris als desenvolupadors de programari. L'ús bàsic de Mercurial és en línia d'ordres i existeix una ordre que és la que gestiona tot el paquet mitjançant opcions anomenada `hg` (en referència al símbol químic del mercuri). Les premisses de disseny de mercurial són el rendiment i l'escalabilitat mitjançant un desenvolupament distribuït i sense necessitat d'un servidor. A més, presenta una gestió robusta d'arxius tant de text com binaris i posseeix capacitats avançades de ramificació i integració i una interfície web integrades.[21]

La seva instal·lació és molt simple amb `apt-get mercurial` i és interessant instal·lar un altre paquet anomenat `Tortoisehg` [23] (`apt-get install tortoisehg`), que és una extensió gràfica a Mercurial i actua com *Workbench*, suportant múltiples repositoris, i permet resoldre conflictes en una forma simple i gràfica. Després de la instal·lació, podrem executar l'ordre `hg version` i `hg debuginstall` on aquesta última ens dirà que hem d'identificar l'usuari i incloure una adreça de correu. Per a això, crearem l'arxiu `more $HOME/.hgrc` amb el següent contingut (la segona part és per a Tortoise):

```
[ui]
username = AdminP <adminp@sysdw.nteum.org>
[extensions]
graphlog =
```

Amb això, ja estem en condicions de començar a treballar en el nostre repositori. Per a usuaris recentment iniciats, és molt recomanable seguir la guia de [22], que mostra el procediment per a gestionar un repositori entre dos usuaris. Una seqüència d'ordres podria ser:

```
Creem un repositori: hg init hello veurem un directori amb un subdirectori .hg
Dins de hello creem un arxiu: echo "Hello World" > hello.txt
Mirem l'estat però veurem que no forma part de repositori: hg status
L'afegim i executem l'ordre anterior. Veurem que ja està dins: hg add hello.txt
Per a publicar-ho en el repositori: hg commit -m "El meu primer hg-hello"
La història la podem veure amb: hg log
Modifiquem i mirem les diferències: echo "Bye..." > hello.txt; hg diff
Podem revertir els canvis: hg revert hello.txt
Afegim informació: echo "by Admin." >> hello.txt
Publiquem els canvis: hg commit -m "Added author."
Preguntem a HG on estan aquests: hg annotate hello.txt
Les anotacions són de molta ajuda per a fixar els errors, ja que ens permeten veure qui i què ha canviat: hg log -r 1 i de manera espacial hg glog o millor en forma gràfica utilitzant Tortoise hg glog
Que ens mostrarà que el canvi és fill de la mateixa branca-autor.
Mirem els parents i tornem enrere (a una versió anterior): hg parents, després hg update 0 si fem hg parents i mirem l'arxiu, veurem que tenim la versió original. Però no ho hem perdut, si volem mirar l'anterior: hg cat -r 1 hello.txt
I recuperar-ho: hg update
Si volem treballar entre dos usuaris que comparteixen sistema d'arxius (podria ser per SSH o HTTP també), l'usuari interessat podrà fer una còpia del repositori:
hg clone ../adminp/hello
Fins i tot es podria fer per correu/USB utilitzant hg bundle per a generar un binari i portarlo a una altra màquina. Aquest nou usuari serà independent d'allò que faci l'altre usuari (adminp).
```

Aquest usuari (remix) podrà fer: `cd hello; echo "My hola" > hello2.txt; echo "by Remix." >> hello2.txt`
 Ho agrega al seu repositori i publica: `hg add hg commit -m "My hello."`
 Després modifiquem el d'Adminp: `echo "not by Remix" >> hello.txt`
 Publiquem: `hg commit -m "Not my file."`
 I podem preguntar si hi ha hagut canvis pel que fa a adminp clone: `hg incoming`.
 I preguntar quins canvis no estan en adminp: `hg outgoing`.
 Remix no té accés al directori d'Adminp, per la qual cosa no pot introduir els canvis però Adminp sí pot llegir-los i agregar-los. Ha de posar en el seu arxiu `.hg/hgrc`

```
[paths]
default = /home/remix/hola
```

I adminp pot observar els canvis amb `hg incoming` i agregar-los amb: `hg pull`. És interessant veure'ls amb Tortoise, que els mostrarà gràficament.
 És interessant que d'una manera simple aquest repositori el podem publicar via web fent en el nostre repositori: `hg serve`
 A partir d'això, es podrà accedir a l'URL `http://sysdw.nteum.org:8000` (o en *localhost*).
 Aquesta solució és simple (però eficient) i temporal. Repositoris més estables hauran de ser publicats utilitzant Apache amb l'*script* `hgweb.cgi/hgwebdir.cgi` indicat en el manual d'instal·lació. [25][26] També és interessant l'ordre `hg-ssh` quan els repositoris s'ofereixen mitjançant SSH.

És interessant completar la informació amb un tutorial [24] i veure les extensions que admet HG [28], i no menys interessants són els paquets (inclosos en Debian) **hgview** (*interactive history viewer*), **mercurial-git** (*git plugin* per a mercurial), **mercurial-server** (*shared Mercurial repository service*) o **eclipse-mercurialeclipse** (integració amb Eclipse).

1.7. Mantis Bug Tracker

Mantis Bug Tracker és un sistema (GPL) de seguiment d'errors a través del web. L'ús més comú de MantisBT és fer un seguiment d'errors o incidències, però també serveix com a eina de seguiment de gestió i administració de projectes. A més del paquet de Mantis, és necessari tenir instal·lat MySQL, Apache Web Server i el mòdul PHP per a Apache. La instal·lació de Mantis és molt simple, `apt-get install mantis`, i durant el procés ens indicarà la contrasenya per a l'usuari administrador (i ens avisarà que és recomanable canviar-la) i demanarà alguna informació més en funció dels paquets que tinguem instal·lats (Apache, MySQL, MariaDB, etc.). A partir d'aquest moment, ens podrem connectar a `http://localhost/mantis/` introduint l'usuari administrador i la contrasenya *root* (es recomana que la primera acció sigui canviar la contrasenya en la secció *MyAccount*). Accedint des de la interfície web (*ítem manage*), es podran crear nous usuaris amb permisos d'usuari per rols (*administrator, viewer, reporter, updater*, etc.), definir els projectes i les categories dins de cada projecte, gestionar els anuncis, informes, registres, tenir una visió general dels projectes i el seu estat, gestionar els documents associats, etc. Es recomana per a crear nous usuaris, noves categories i després projectes assignats a aquestes categories. A partir d'aquest moment, es poden inserir incidències, assignant-les i gestionant-les mitjançant la interfície. Si tenim instal·lada MariaDB, ens podrà donar un error similar a `mysql_connect()`: *Headers and client*

library minor version mismatch i haurem d'actualitzar els *headers i drivers* de PHP `apt-get install php5-mysqlnd`.

Existeix un ampli conjunt d'opcions que es poden canviar modificant l'arxiu `/usr/share/mantis/www/config_inc.php` [30], així, per a canviar la llengua de l'entorn, s'edita i afegeix

```
$g_language_choices_arr = array( 'english', 'spanish'); $g_default_language  
= 'spanish';
```

i perquè la pàgina principal del Mantis Bug Tracker sigui automàticament el propi Bug Tracker i es permeti un accés anònim als projectes públics:

1) Crear un compte d'usuari, per exemple *anonymous* o *guest*, deixant en blanc el *Real Name*, *Email=anonymous@localhost* (o deixar en blanc si es posa `$g_allow_blank_email = ON`), *Access Level = viewer* o *reporter* (depenent dels que es vulguin) *Enabled = true* i *Protected = true*.

2) Després, s'ha de modificar en l'arxiu anterior (`config_inc.php`) posant les següents variables:

```
$g_allow_anonymous_login = ON;  
$g_anonymous_account = 'anonymous'
```

i, opcionalment, per a deixar en blanc les adreces de correu:

```
$g_allow_blank_email = ON
```

*http://www.mantisbt.org/wiki/doku.php/mantisbt:mantis_recipies
Per a més configuracions o integracions amb altres paquets, consulteu el manual [29] o accediu a la pàgina de la *wiki* del projecte* [30].

Activitats

1. Definiu en PostgreSQL una base de dades que tingui almenys 3 taules amb 5 columnes (de les quals 3 han de ser numèriques) en cada taula. Genereu una llista ordenada per cada taula/columna. Genereu una llista ordenada pel major valor de la columna X de totes les taules. Canvieu el valor numèric de la columna i amb el valor numèric de la columna Z + el valor de la columna W/2.
2. Feu el mateix que amb l'exercici anterior, però amb MySQL o amb MariaDB.
3. Utilitzant les taules de World <http://pgfoundry.org/projects/dbsamples/> obtingueu la població mitjana de les poblacions de les ciutats entre 10.000 i 30.000 habitants, i un percentatge de quina és la llengua més parlada i la menys parlada en relació amb els habitants.
4. Configureu el CVS per a fer tres revisions d'un directori on hi ha 4 arxius .c i un Makefile. Feu una ramificació (*branch*) d'un arxiu i després barregeu-lo amb el principal.
5. Simuleu la utilització d'un arxiu concurrent amb dos terminals de Linux i indiqueu la seqüència de passos per a fer que dues modificacions alternes de cada usuari quedin reflectides sobre el repositori CVS.
6. Feu el mateix exercici anterior, però un dels usuaris ha de connectar-se al repositori des d'una altra màquina.
7. Feu novament els tres exercicis anteriors, però en Subversion.
8. Creeu un repositori sobre Git i feu-lo visible a través del web, de tal manera que dos usuaris de la mateixa màquina puguin modificar els arxius i actualitzar el repositori.
9. Repetiu el pas anterior amb Mercurial.
10. Instal·leu Mantis, genereu 3 usuaris, 2 categories, 2 projectes i 3 incidències per a cada usuari, i gestioneu-les a través de la interfície. Genereu un usuari anònim que pugui accedir a un projecte públic com a *reporter*, però no a un de privat.

Bibliografia

- [1] *Big Data - Infografia*.
<<http://i2.wp.com/unadocenade.com/wp-content/uploads/2013/05/Infograf%C3%ADa-Big-Data.jpg>>
- [2] M. Gibert, O. Pérez. *Bases de datos en PostgreSQL*.
<http://ocw.uoc.edu/computer-science-technology-and-multimedia/bases-de-datos/bases-de-datos/P06_M2109_02152.pdf>
- [3] *PostgreSQL*.
<<http://www.postgresql.org>>
- [4] *PostgreSQL. The SQL Language*.
<<http://www.postgresql.org/docs/9.1/interactive/tutorial-sql.html>>
- [5] *PgAdmin*.
<<http://www.pgadmin.org>>
- [6] *PgpPGAdmin*.
<<http://phppgadmin.sourceforge.net/>>
- [7] *Documentación de MySQL*.
<http://dev.mysql.com/usingmysql/get_started.html>
- [8] *MySQL Administrator*.
<<http://www.mysql.com/products/workbench/>>
- [9] *MariaDB*.
<<https://mariadb.org/>>
- [10] *Setting up MariaDB Repositories*.
<<https://downloads.mariadb.org/mariadb/repositories/>>
- [11] *SQLite*.
<<http://www.sqlite.org/>>

- [12] *SQLite Database Browser*.
<<https://github.com/sqlitebrowser/sqlitebrowser>>
- [13] **Cederqvist**. *Version Management with CVS*.
<<http://www.cvshome.org>>
- [14] *Llibre sobre Subversion*.
<<http://svnbook.red-bean.com/nightly/es/index.html>>
(versió en espanyol)
- [15] *Subversion Documentation*.
<<http://subversion.tigris.org/servlets/ProjectDocumentList>>
bibitemsub *Subversion*.
<<http://subversion.apache.org/>>
- [16] *Múltiples repositoris amb Subversion*.
<<http://www.debian-administration.org/articles/208>>
- [17] *Git. Fast Control Version system*.
<<http://git-scm.com/>>
- [18] *Instalar Git sobre Debian*.
<<http://linux.koolsolutions.com/2009/08/07/learn-git-series-part-1-installing-git-on-debian/>>
- [19] **L. Hernández**. *Servidor Git + Gitolite + Gitweb sobre Debian*.
<<http://leninmhs.wordpress.com/2014/01/19/git-gitolite-gitweb/>>
- [20] **D. Mühlbachler**. *How-To: Install a private Debian Git server using gitolite and GitLabHQ*.
<<http://blog.muehlbachler.org/2012/01/how-to-install-a-private-debian-git-server-using-gitolite-and-gitlabhq/>>
- [21] *Mercurial*.
<<http://mercurial.selenic.com/>>
- [22] *Basic Mercurial*.
<<http://mercurial.aragost.com/kick-start/en/basic/>>
- [23] *TortoiseHg Docs*.
<<http://tortoisehg.bitbucket.org/docs.html>>
- [24] *HgInit: a Mercurial tutorial*.
<<http://hginit.com/>>
- [25] *Publishing Repositories with hgwebdir.cgi*.
<<http://mercurial.selenic.com/wiki/HgWebDirStepByStep>>
- [26] *Publishing Mercurial Repositories*.
<<http://mercurial.selenic.com/wiki/PublishingRepositories>>
- [27] *Remote Repositories*.
<<http://mercurial.aragost.com/kick-start/en/remote/>>
- [28] *Using Mercurial Extensions*.
<<http://mercurial.selenic.com/wiki/UsingExtensions>>
- [29] *Documentació del projecte Mantis*.
<<http://www.mantisbt.org/documentation.php>>
- [30] *Mantis Bug Tracker Wiki*.
<<http://www.mantisbt.org/wiki/doku.php>>
- [31] *Mòduls de Webmin*.
<<http://doxfer.webmin.com/Webmin>>
- [32] **Ibiblio.org** (2010). *Linux Documentation Center*.
<<http://www.ibiblio.org/pub/Linux/docs/HOWTO/>>
- [33] **Mourani, G.** (2001). *Securing and Optimizing Linux: The Ultimate Solution*. Open Network Architecture, Inc.
- [34] *TkCVS: Tcl/Tk-based graphical interface to the CVS*.
<<http://www.twobarleycorns.net/tkcv.html>>
- [35] *Cervisia - CVS Frontend*.
<<http://www.kde.org/applications/development/cervisia/>>

Administració de seguretat

Josep Jorba Esteve

PID_00212467

Índex

Introducció	5
Objectius	7
1. Tipus i mètodes dels atacs	9
1.1. Tècniques utilitzades en els atacs	12
1.2. Contramesures	20
2. Seguretat del sistema	25
3. Seguretat local	26
3.1. <i>Bootloaders</i>	26
3.2. Contrasenyes i <i>shadow</i> s	28
3.3. Bits <i>sticky</i> i <i>setuid</i>	30
3.4. Habilitació de <i>hosts</i>	30
3.5. Mòduls PAM	31
3.6. Alteracions del sistema	33
3.7. Recursos limitats, <i>cgroups</i> i <i>chroot</i>	33
3.8. Protecció d'executables mitjançant Hardening-Wrapper	38
4. SELinux	41
4.1. Arquitectura	44
4.2. Crítica	47
4.3. Algunes alternatives	48
5. Seguretat en xarxa	50
5.1. Client de serveis	50
5.2. Servidor: <i>inetd</i> i <i>xinetd</i>	50
6. Eines de seguretat: detecció de vulnerabilitats i intrusions	53
6.1. OpenVAS	57
6.2. Denyhosts	60
6.3. Fail2ban	62
7. Protecció mitjançant filtratge (<i>TCP wrappers</i> i tallafocs) ..	65
7.1. Tallafocs	66
7.2. Netfilter: <i>iptables</i>	67
7.3. Paquets per a la gestió de tallafocs en les distribucions	72
7.4. Netfilter: <i>nftables</i>	74

7.5. Consideracions	76
8. Anàlisi de registres	78
9. Taller: anàlisi de la seguretat mitjançant eines	80
Resum	88
Activitats	89
Bibliografia	90

Introducció

El salt tecnològic que s'ha produït des dels sistemes d'escriptori aïllats fins als sistemes actuals integrats en xarxes locals i Internet, ha dut una nova dificultat a les tasques habituals de l'administrador: el control de la seguretat dels sistemes.

La seguretat és un camp complex, en el qual es barregen tècniques d'anàlisi amb altres de detecció o de prevenció dels possibles atacs. Les tècniques que cal utilitzar són tant computacionals com relacionades amb altres camps, com ara l'anàlisi de factors psicològics, pel que fa al comportament dels usuaris del sistema o a les possibles intencions dels atacants.

Els atacs poden provenir de moltes fonts i afectar des d'una aplicació o servei fins a algun usuari, a tots o el sistema informàtic sencer.

Els possibles atacs poden canviar el comportament dels sistemes, fins i tot "fer-los caure" (és a dir, inutilitzar-los), o donar una falsa impressió de seguretat, que pot ser difícilment perceptible. Podem trobar-nos amb atacs d'autenticació (obtenir accés per part de programes o usuaris prèviament no habilitats), escoltes (redirigir o punxar els canals de comunicació i les dades que hi circulen) o substitució (de programes, màquines, comunicacions o usuaris per d'altres, sense que es notin els canvis).

Seguretat absoluta

La seguretat absoluta no existeix. Una falsa impressió de seguretat pot ser tan perjudicial com no tenir-ne. L'àrea de la seguretat és molt dinàmica i cal mantenir actualitzats constantment els coneixements.

Una idea clara que cal tenir sempre present és que és impossible aconseguir una seguretat del 100%.

Les tècniques de seguretat són una arma de doble tall, que fàcilment poden donar-nos una falsa impressió de control del problema. La seguretat actual és un problema ampli, complex i, el que és més important, dinàmic. Mai no podem esperar o assegurar que la seguretat estigui garantida en un determinat sistema, sinó que amb força probabilitats serà una de les àrees a les quals l'administrador haurà de dedicar més temps i mantenir actualitzats els seus coneixements sobre el tema.

En aquest mòdul examinarem diferents problemàtiques amb què podem trobar-nos, com podem verificar i prevenir parts de la seguretat local i en entorns de xarxa. També examinarem tècniques de detecció d'intrusions i algunes eines bàsiques que ens poden ajudar en el control de la seguretat.

També cal esmentar que en aquest mòdul només podem fer una breu introducció a alguns dels aspectes que intervenen en la seguretat d'avui en dia. Per a qualsevol aprenentatge real amb més detall, es recomana consultar la bibliografia disponible, i els manuals associats als productes o eines comentats.

Objectius

En aquest mòdul mostrarem els continguts i les eines procedimentals per aconseguir els objectius següents:

1. Conèixer els principals tipus i mètodes d'atacs a la seguretat dels sistemes, així com algunes contramesures bàsiques i eines útils per al seu tractament.
2. Saber fer un seguiment de la seguretat local i en xarxa en els sistemes GNU/Linux.
3. Conèixer la metodologia d'ús d'eines de detecció d'intrusions.
4. Saber elaborar mesures de prevenció mitjançant filtratge de serveis o paquets *wrappers* i tallafocs (*firewalls*).
5. Conèixer les eines de seguretat disponibles per a controlar accessos i permisos, a nivell de MAC (*Mandatory Access Control*).

1. Tipus i mètodes dels atacs

La seguretat computacional, en administració, pot ser entesa com el procés que ha de permetre a l'administrador del sistema prevenir i detectar usos no autoritzats d'aquest sistema. Les mesures de prevenció ajuden a aturar els intents d'usuaris no autoritzats (els coneguts com a **intrusos**) per a accedir a qualsevol part del sistema. La detecció ajuda a descobrir quan es van produir aquests intents o, en el cas de dur-se a terme, establir les barreres perquè no es repeteixin i poder recuperar el sistema si ha estat trencat o compromès.

Els intrusos (també coneguts col·loquialment com a *hackers*, *crackers*, *furoners*, *atacants* o *pirates*) normalment volen controlar el sistema, o bé per a causar funcionaments erronis, per a malmetre el sistema o les seves dades, per a guanyar recursos a la màquina o bé, simplement, per a llançar atacs contra altres sistemes i així protegir la seva veritable identitat i ocultar l'origen real dels atacs. També hi ha la possibilitat que l'atac es produeixi per a examinar o robar la informació del sistema, el pur espionatge de les accions del sistema o per a causar danys físics a la màquina, que poden consistir a formatar el disc, canviar dades, esborrar o modificar programari crític, etc.

Respecte als intrusos, cal establir algunes diferències que no solen estar gaire clares en els termes col·loquials. Normalment amb **hacker** [Him01] ens referim a una persona amb grans coneixements d'informàtica, més o menys apassionada pels temes de programació i seguretat informàtica i que, normalment sense finalitat malèvola, utilitza els coneixements per a protegir-se a ell mateix o protegir tercers, introduir-se en xarxes per a demostrar-ne les fallades de seguretat i, en alguns casos, per a obtenir el reconeixement públic de les seves habilitats. Un exemple seria la mateixa comunitat GNU/Linux, que deu molt als seus *hackers*, ja que cal entendre el terme *hacker* com a expert en uns temes computacionals, més que com a intrús que afecta la seguretat.

D'altra banda, trobaríem els pirates o **crackers**. Aquí és quan s'utilitza el terme, de manera més o menys despectiva, per a referir-se a aquells que utilitzen les habilitats per a malmetre (o destrossar) sistemes, només per a obtenir fama, per motius econòmics, per ganes de causar dany o simplement per a molestar, per motius d'espionatge tecnològic, com a actes de ciberterrorisme, etc.

Així mateix, es parla de *hacking* o *cracking* o pirateig, respectivament, quan ens referim a tècniques d'estudi, detecció i protecció de la seguretat, o al contrari, a tècniques destinades a causar dany trencant la seguretat dels sistemes.

Altres termes, més comuns actualment per a diferenciar *hackers* (quan el terme s'usa de manera genèrica) en la comunitat de seguretat (a causa del mal

ús dels anteriors), són *black hat* per a referir-se a un *hacker* que viola la seguretat informàtica per raons més enllà de la malícia o per al benefici personal. Són la personificació d'allò que s'entén clàssicament per criminal informàtic. Els *hackers black hat* entren a xarxes segures per a destruir les dades o fer-les inutilitzables per a aquells que tinguin accés autoritzat. Normalment el procés que segueixen consta de l'elecció dels objectius, la recopilació d'informació i recerca i, finalment, l'execució. D'altra banda, es coneixen com *white hat* aquells *hackers* que trenquen la seguretat per raons no malicioses, potser per posar a prova els seus propis sistemes, desenvolupant programari de seguretat o analitzant la seguretat de sistemes o organitzacions sota un acord contractual. També s'acostumen a denominar *hackers ètics*. Uns altres de més curiosos són els denominats *gray hat*, que actuen com els *black hat* atacant sistemes, però s'ofereixen per a arreglar-los a un mòdic preu. O els *script kiddies*, que són inexperts i aprofiten eines automatitzades empaquetades i efectuades per altres, generalment sense cap (o poca) comprensió dels conceptes subjacents i moltes vegades sense ser amb prou feines conscients dels danys que poden provocar.

Malauradament, obtenir accés a un sistema (desprotegit o parcialment segur) és bastant més fàcil del que sembla a primera vista. Els intrusos descobreixen permanentment noves **vulnerabilitats** (anomenades de vegades *forats de seguretat*), que els permeten introduir-se en les diferents capes de programari (aplicacions, serveis o parts del sistema operatiu). Així, definirem una vulnerabilitat com l'error d'un programa (o capa de programari) que permet, mitjançant la seva explotació, violar la seguretat d'un sistema informàtic. D'altra banda, un *exploit* és un programari, conjunt de dades o seqüència d'ordres que, traient profit de vulnerabilitats, permet causar comportaments inesperats o no anticipats, a programari, maquinari o dispositius electrònics.

La complexitat cada vegada més gran del programari (i del maquinari) fa que augmenti la dificultat per a provar de manera raonable la seguretat dels sistemes informàtics. L'ús habitual dels sistemes GNU/Linux en xarxa, sigui a la mateixa Internet o en xarxes pròpies amb tecnologia TCP/IP com les intranets, ens duu a exposar els nostres sistemes, com a víctimes, a atacs de seguretat [Bur02, Fen02, Line].

El primer que cal fer és trencar el mite de la seguretat informàtica: simplement no existeix. El que sí que podem aconseguir és un cert grau de seguretat que ens faci sentir segurs dins de certs paràmetres. Però com a tal, això només és una percepció de seguretat, i com totes les percepcions, pot ser falsa i d'això podem adonar-nos-en en el darrer moment, quan ja tinguem els nostres sistemes afectats i compromesos. La conclusió lògica és que la seguretat informàtica exigeix un esforç important de constància, realisme i aprenentatge pràcticament diari.

Hem de ser capaços d'establir en els nostres sistemes unes polítiques de seguretat que ens permetin prevenir, identificar i reaccionar davant dels possibles

atacs. I tenir present que la sensació que puguem tenir de seguretat no és res més que això, una sensació. Per tant, no s'ha de descuidar cap de les polítiques implementades i cal mantenir-les al dia, de la mateixa manera que els nostres coneixements del tema.

Els possibles atacs són una amenaça constant als nostres sistemes i poden comprometre'n el funcionament, així com les dades que gestionem. Davant de tot això, sempre hem de definir una certa política de requisits de seguretat sobre els nostres sistemes i dades. Les amenaces que podem patir podrien afectar els aspectes següents:

Amenaces

Les amenaces afecten la confidencialitat, la integritat o l'accessibilitat dels nostres sistemes.

- **Confidencialitat:** la informació ha de ser accessible només a aquells que hi estiguin autoritzats; estem responenent a la pregunta *qui podrà accedir a la informació?*
- **Integritat:** la informació només podrà ser modificada per aquells que hi estiguin autoritzats: què es podrà fer amb la informació?
- **Accessibilitat:** la informació ha d'estar disponible per als qui la necessitin i quan la necessitin, si hi estan autoritzats: de quina manera, i quan, es podrà accedir a la informació?

Passem a esmentar una determinada classificació (no exhaustiva) dels tipus d'atacs habituals que podem patir:

- **Autenticació:** atacs en els quals es falsifica la identitat del participant, de manera que obté accés a programes o serveis als quals en principi no tenia accés.
- **Intercepció** (o escolta): mecanisme pel qual tercers intercepten dades quan aquestes no anaven dirigits a ells.
- **Falsificació** (o reemplaçament): substitució d'alguns dels participants, tant màquines com programari o dades, per altres de falsos.
- **Robatori de recursos:** ús dels nostres recursos sense autorització.
- **Vandalisme:** al cap i a la fi, sol ser bastant comú l'existència de mecanismes que permeten interferir en el funcionament adequat del sistema o dels serveis i causar molèsties parcials i l'aturada o cancel·lació de recursos.

Els mètodes utilitzats i les tècniques necessàries poden variar molt (encara més, cada dia es creen novetats) i ens obliguen, com a administradors, a estar en contacte permanent amb el camp de la seguretat per a conèixer a què ens podem enfrontar diàriament.

Pel que fa a on es produeix l'atac, hem de tenir clar què pot fer-se o quin serà l'objectiu dels mètodes:

Finalitat dels atacs

Els atacs poden tenir finalitats destructives, inhabilitadores o d'espionatge dels nostres components (tant maquinari com programari o sistemes de comunicació).

- **Maquinari.** Quant a això, l'amenaça està directament sobre l'accessibilitat: què podrà fer algú que tingui accés al maquinari? En aquest cas normalment necessitarem mesures "físiques", com ara controls de seguretat per a accedir als locals on estiguin situades les màquines, per a evitar problemes de robatori o ruptura de l'equip a fi d'eliminar-ne el servei. També pot comprometre's la confidencialitat i la integritat si l'accés físic a les màquines permet utilitzar alguns dels dispositius, com les unitats de disc (extraïbles o no), l'arrencada de les màquines o l'accés a comptes d'usuari que podrien estar oberts.
- **Programari.** Si l'accessibilitat es veu compromesa en un atac, es poden esborrar o inutilitzar programes, denegant-ne l'accés. En cas de confidencialitat, pot provocar còpies no autoritzades de programari. En integritat podria alterar-se el funcionament per defecte del programa, perquè falli en algunes situacions o bé perquè faci tasques que puguin ser interessants per a l'atacant, o podria simplement comprometre la integritat de les dades dels programes: fer-los públics, alterar-los o simplement robar-los.
- **Dades,** ja siguin estructurades, com en els serveis de base de dades, gestió de versions (com cvs) o simples arxius. Mitjançant atacs que amenacin l'accessibilitat, aquestes dades poden ser destruïdes o eliminades, de manera que s'hi denegui l'accés. En el cas de la confidencialitat, estariem permetent lectures no autoritzades i la integritat es veuria afectada quan es produïssin modificacions o creació de noves dades.
- **Canal de comunicació** (a la xarxa, per exemple). Per als mètodes que afecten l'accessibilitat, ens pot provocar la destrucció o l'eliminació de missatges i impedir l'accés a la xarxa. En confidencialitat, es pot donar la lectura i observació del trànsit de missatges, des de la màquina o cap a la màquina. I respecte a la integritat, podem veure'ns afectats per qualsevol modificació, retard, reordenació, duplicació o falsificació dels missatges entrants o sortints.

1.1. Tècniques utilitzades en els atacs

Els mètodes utilitzats són múltiples i poden dependre d'un element (maquinari o programari) o de la seva versió. Per tant, cal mantenir actualitzat el programari per les correccions de seguretat que vagin apareixent i seguir les indicacions del fabricant o distribuïdor per a protegir l'element, a més d'utilitzar les mesures de seguretat activa que puguem aplicar.

Malgrat això, normalment sempre hi ha tècniques o mètodes “de moda” del moment actual. Algunes indicacions breus d’aquestes tècniques d’atac (actuals) són:

- **Explotació de forats (*bug exploits*):** es tracta de l’explotació de forats o errors [Cerb, Ins, San] d’un maquinari, programari, servei, protocol o el mateix sistema operatiu (per exemple, en el nucli o *kernel*) i, normalment, d’alguna de les seves versions en concret. En general, qualsevol element informàtic és més o menys propens a errors en la seva concepció o simplement a coses que no s’han tingut en compte o previst. Periòdicament, es descobreixen forats (de vegades s’anomenen *holes*, *exploits* o simplement *bugs*), que poden ser aprofitats per un atacant per a trencar la seguretat dels sistemes. Solen utilitzar-se tècniques d’atac genèriques, com les que s’expliquen a continuació, o bé tècniques particulars per a l’element afectat. Cada element tindrà un responsable, tant fabricant, desenvolupador, distribuïdor com la comunitat GNU/Linux, de produir noves versions o pedaços per a tractar aquests problemes. Nosaltres, com a administradors, tenim la responsabilitat d’estar informats i de mantenir una política d’actualització responsable per a evitar els riscos potencials d’aquests atacs. En cas que no hi hagi solucions disponibles, també podem estudiar la possibilitat d’utilitzar alternatives a l’element, o bé inhabilitar-lo fins que tinguem solucions.
- **Virus:** es tracta de programes normalment annexos a d’altres i que utilitzen mecanismes d’autocòpia i transmissió. Són habituals els virus adjunts a programes executables, a missatges de correu electrònic o incorporats en documents o programes que permeten algun llenguatge de macros (no verificat). Són potser la pesta més important de seguretat d’avui en dia en alguns sistemes.

Els sistemes GNU/Linux estan protegits gairebé totalment (n’hi ha diversos casos, com també proves de concepte) contra aquests mecanismes vírics per diverses raons: en els programes executables, tenen un accés molt limitat al sistema, en particular al compte de l’usuari, amb excepció de l’usuari *root*, el compte del qual s’hauria de restringir als usos adequats, i també hauria de limitar-se o evitar-se completament l’ús com a usuari del sistema; el correu no sol utilitzar llenguatges de macros no verificats (com en el cas de l’Outlook i Visual Basic Script a Windows, que en el seu dia, van ser un aport important de forats d’entrada per a virus), mentre que en el cas dels documents, estem en condicions semblants, ja que no suporten llenguatges de macros no verificats (com el VBA en Microsoft Office). Tot i això, cal comentar que diversos aspectes d’aquest tipus comencen a tenir una certa importància, ja que alguns dels paquets d’ofimàtica per a GNU/Linux utilitzen ja llenguatges de macros, i els clients de correu cada vegada incorporen més suport d’HTML encastat amb suport de JavaScript, una de les fonts amb més problemes, com veurem en alguns apartats següents. Que aquests problemes no hagin estat explotats (o només mínimament) és no-

Evolució de les tècniques d’atac

Les tècniques dels atacs són molt variades i evolucionen constantment pel que fa als detalls tecnològics usats.

més una qüestió DE l'ús d'una plataforma, com GNU/Linux. A mesura que creix l'ús, també es fa més atractiva per als atacants.

En qualsevol cas, caldrà fer atenció al que pugui passar en un futur, ja que podrien sorgir alguns virus específics per a GNU/Linux aprofitant alguns errors o forats dels components del sistema. Un punt que sí que cal tenir en compte és el dels sistemes de correu, ja que si bé nosaltres no generarem virus, sí que podem arribar a transmetre'ls; per exemple, si el nostre sistema funciona com a *router*, *relay*, o simplement com a servidor de correu, podrien arribar missatges amb virus i aquests podrien ser enviats a d'altres. Aquí es pot aplicar alguna política de detecció i filtratge de virus, si en els nostres sistemes hi ha entorns Windows, en els quals es poden propagar virus externs que hàgim rebut. Un altre problema molt important que podria entrar dins de la categoria de virus són els correus brossa (*spam*), que si bé no solen ser utilitzats com a elements atacants (tot i que poden combinar-se amb altres tècniques, com intents de pesca), sí que podem considerar-los com un problema per la “virulència” de la seva aparició i pel cost econòmic que poden representar (pèrdues de temps i de recursos).

- **Cucs (*worms*):** normalment es tracta d'un tipus de programes que aprofiten algun forat del sistema per a executar codi sense permís. Solen ser utilitzats per a aprofitar recursos de la màquina, com l'ús de CPU, quan es detecta que el sistema no funciona o no està en ús o, si són malintencionats, amb l'objectiu de robar recursos o utilitzar-los per a aturar o bloquejar el sistema. També solen utilitzar tècniques de transmissió i replicació.
- **Troians o cavalls de Troia (*trojans* o *Trojan horses*):** programes útils que incorporen alguna funcionalitat però n'oculten d'altres, que són les utilitzades per a obtenir informació del sistema o comprometre'l. Un cas particular pot ser el dels codis de tipus mòbil en aplicacions web, com els Java, JavaScript o ActiveX; aquests normalment demanen consentiment per a executar-se (ActiveX a Windows) o tenen models limitats de què poden fer (Java, JavaScript). Però, com qualsevol programari, també poden tenir forats, i són un mètode ideal per a transmetre cavalls de Troia.
- **Portes secretes (*back doors*):** és un mètode d'accés amagat en un programa que pot utilitzar-se per a donar accés al sistema o a les dades manejades sense que ho sapiguem. Altres efectes poden ser canviar la configuració del sistema o permetre la introducció de virus. El mecanisme que es fa servir pot ser des d'estar inclòs en algun programari comú fins a venir en un troià que produeixi portes secretes.
- **Bombes lògiques:** programa incrustat en un altre que comprova que es donin determinades condicions (temporals, accions de l'usuari, etc.) per a activar-se i emprendre accions no autoritzades.

Cuc Morris

Un dels primers cucs, el conegut com a Cuc Morris, va ser molt important perquè va fer de la seguretat un tema important en uns moments en què hi havia certa innocència en aquests temes. Vegeu http://en.wikipedia.org/wiki/Morris_worm.

- **Enregistradors de teclat (*keyloggers*):** es tracta d'un programa especial que es dedica a segrestar les interaccions amb el teclat o el ratolí de l'usuari. Poden ser programes individuals o bé cavalls de Troia incorporats en altres programes. Normalment, necessitarien introduir-se en un sistema obert al qual es tingués accés (tot i que, cada vegada més sovint, poden anar incorporats en troians que s'instal·lin). La idea és captar qualsevol introducció de tecles, de manera que es capturin contrasenyes (per exemple, les bancàries), la interacció amb aplicacions, els llocs visitats per la xarxa, els formularis emplenats, etc.
- **Escaneig de ports (*port scanning*):** més que un atac, seria un pas previ, que consistiria en la recollida de possibles objectius. Bàsicament, consisteix a utilitzar eines que permetin examinar la xarxa a la recerca de màquines amb ports oberts, tant TCP, UDP com altres protocols que indiquen la presència d'alguns serveis. Per exemple, escanejar màquines buscant el port 80 TCP indica la presència de servidors web, dels quals podem obtenir informació sobre el servidor i la versió que utilitzen i així aprofitar-nos de vulnerabilitats conegudes.
- **Detectors (*sniffers*):** permeten la captura de paquets que circulen per una xarxa. Amb les eines adequades podem analitzar comportaments de màquines: quines són servidors i quines són clients, quins protocols s'utilitzen i, en molts casos, obtenir contrasenyes de serveis no segurs. Al principi van ser molt utilitzats per a capturar contrasenyes de Telnet, rsh, rcp, FTP, etc., serveis no segurs que ja no s'haurien d'utilitzar (cal fer servir, en canvi, les versions segures: ssh, scp, sftp). Tant els detectors com els escàners de ports no són necessàriament eines d'atac, ja que també poden servir per a analitzar les nostres xarxes i detectar errors, o simplement per a analitzar el nostre trànsit. Normalment, un intrús sol utilitzar tant les tècniques d'escaneig com les dels detectors amb l'objectiu de trobar les vulnerabilitats del sistema, o bé per a conèixer dades d'un sistema desconegut (escàners) o bé per a analitzar-ne la interacció interna (detectors).
- **Segrest (*hijacking*):** són tècniques que intenten col·locar una màquina de manera que intercepti o reproduïxi el funcionament d'algun servei en una altra màquina de la qual ha punxat la comunicació. Solen ser habituals els casos per a correu electrònic, transferència de fitxers o web. Per exemple, en el cas web, es pot capturar una sessió i reproduir el que l'usuari està fent, pàgines visitades, interacció amb formularis, etc. En alguns casos pot ser a causa de segrestos en l'àmbit de xarxa, perquè es capturen o escolten paquets de les comunicacions, o pot produir-se un segrest fora de línia (*offline*) mitjançant l'execució arbitrària de codi de *scripts*. Per exemple, és habitual, en sistemes de correu electrònic per web (*webmail*) o en aplicacions web que incloguin l'ús d'HTML en algunes de les seves finestres, que el sistema permeti injectar codi JavaScript o PHP (o altres llenguatges d'*script* similars) i això, si no es controla o limita de manera correcta, pot provocar el robatori de dades de l'usuari (o bé els identificadors de sessió o bé les ga-

letes –*cookies*– utilitzades), cosa que permet a l'atacant reproduir o segrestar *a posteriori* les sessions de correu o d'aplicació web sense necessitat de cap contrasenya, i així segrestar la sessió i la identitat de l'usuari.

- **Desbordament (*buffer overflows*):** tècnica bastant complexa que aprofita errors de programació en les aplicacions. La idea bàsica és aprofitar desbordaments (*overflows*) de memòria intermèdia (*buffers*) de l'aplicació, ja siguin cues, matrius, vectors, etc. Si no se'n controlen els límits, un programa atacant pot generar un missatge o dada més gran de l'esperat i provocar fallades mitjançant l'escriptura per sobre dels límits permesos per l'estructura de dades i, així, sobreescriure memòria adjacent. Per exemple, en moltes aplicacions C amb *buffers* mal escrits, en matrius, si sobrepassem el límit podem provocar una sobrescriptura del codi del programa, la qual cosa provoca un funcionament incorrecte o la caiguda del servei o màquina. Encara més, una variant més complexa permet incorporar en l'atac trossos de programa (compilats en C o bé *scripts* de l'interpret d'ordres) que poden permetre l'execució de qualsevol codi que l'atacant vulgui introduir.

Algunes variacions d'aquesta tècnica es poden aprofitar per a atacar de manera similar altres parts del codi executable d'un programa, com per exemple la pila del programa (parlem llavors de *stack overflows*) o la gestió de les peticions dinàmiques de memòria que faci l'executable (*heap overflows*). L'alteració tant de la pila com de la memòria dinàmica també pot permetre a l'atacant l'execució de codi arbitrari afegit o la caiguda de l'executable o servei.

- **Denegació de servei, *denial of service*, DoS:** aquest tipus d'atac provoca que la màquina caigui o que se sobrecarreguin un o més serveis, de manera que no siguin utilitzables. Una altra tècnica és la DDoS (*distributed DoS*, DoS distribuïda), que es basa a utilitzar un conjunt de màquines distribuïdes perquè produeixin l'atac o sobrecàrrega de servei. Aquest tipus d'atacs se solen solucionar amb actualitzacions del programari, i amb una correcta configuració dels paràmetres de control de càrrega del servei, ja que normalment es veuen afectats aquells serveis que no van ser pensats per a una càrrega de treball determinada i no se'n controla la saturació. Els atacs DoS i DDoS són bastant utilitzats en atacs a llocs web o servidors DNS, que es veuen afectats per vulnerabilitats dels servidors, per exemple, de versions concretes d'Apache o BIND. Un altre aspecte que es pot tenir en compte és que el nostre sistema també podria ser usat per a atacs de tipus DDoS, mitjançant control, ja sigui a través d'una porta secreta o d'un troià que formi part d'una *botnet*, per exemple, que disposi d'una sèrie de programes a múltiples màquines interconnectades, les quals podrien trobar-se en un estat latent i participar *a posteriori* en atacs DDoS a demanda.

Un exemple d'aquest atac (DoS) bastant senzill és el conegut com a *SYN flood*, que tracta de generar paquets TCP que obren una connexió, però ja no hi fan res més, simplement la deixen oberta. Això consumeix recursos del sistema en estructures de dades del nucli i recursos de connexió per xar-

xa. Si es repeteix aquest atac centenars o milers de vegades, s'aconsegueix ocupar tots els recursos sense utilitzar-los, de manera que quan alguns usuaris vulguin utilitzar el servei, els sigui denegat perquè els recursos estan ocupats. Un altre cas conegut és el bombardeig de correu (*mail bombing*) o, simplement, el reenviament de correu (normalment amb emissor fals) fins que se saturen els comptes de correu i el sistema de correu cau o es torna tan lent que és inutilitzable. Aquests atacs són, en certa manera, de realització senzilla amb les eines adequades, i no tenen una solució fàcil, ja que s'aprofiten del funcionament intern dels protocols i serveis; en aquests casos hem de prendre mesures de detecció i control posterior, així com posar límits als paràmetres i la càrrega de serveis involucrats en aquests problemes.

- **Falsejament d'identitat (*spoofing*):** les tècniques de falsejament d'identitat engloben diversos mètodes (normalment complexos) per a falsificar tant la informació com els participants en una transmissió (origen i/o destinació). Alguns mètodes de falsejament d'identitat són els següents:
 - Falsejament d'IP (*IP spoofing*), falsificació d'una màquina, de manera que generi trànsit fals o escolti trànsit que anava dirigit a una altra màquina. Combinat amb altres atacs, pot saltar-se fins i tot la protecció de tallafocs.
 - Falsejament d'ARP (*ARP spoofing*), tècnica complexa (utilitza un DDoS) que intenta falsificar les adreces de fonts i destinataris d'una xarxa, mitjançant atacs de les memòries cau d'ARP que posseeixen les màquines, de manera que se substitueixin les adreces reals per d'altres en diversos punts d'una xarxa. Aquesta tècnica permet saltar-se tot tipus de proteccions, tallafocs inclosos, però no és una tècnica senzilla.
 - Correu electrònic. És potser el més senzill. Es tracta de generar continguts de correus falsos, tant pel contingut com per l'adreça d'origen. En aquest tipus són bastant utilitzades les tècniques del que s'anomena *enginyeria social*, que bàsicament intenta enganyar l'usuari d'una manera creïble. Un exemple clàssic són els correus falsos de l'administrador del sistema, o bé del banc on tenim el nostre compte corrent, en què s'esmenten problemes amb la gestió dels nostres comptes i ens demanen enviar informació confidencial o la contrasenya per a solucionar-los, o se'ns demana que es canviï la contrasenya per una altra de concreta. Sorprenentment, aquesta tècnica (també coneguda com a *pesca* o *phishing*) aconsegueix enganyar un nombre considerable d'usuaris. Fins i tot amb enginyeria social de mètodes senzills: algun pirata famós comentava que el seu mètode preferit era el telèfon. Com a exemple, exposem el cas d'una empresa de certificació (Verisign), de la qual els pirates van obtenir la signatura privada de programari de Microsoft simplement amb una trucada, esmentant que trucaven de part de l'empresa, que se'ls havia presentat un problema i que tornaven a necessitar la clau. Resumint, un grau molt alt de seguretat in-

Enllaços d'interès

Sobre SYN flood, vegeu:

<http://www.cert.org/advisories/CA-1996-21.html>.

Sobre bombardeig de correu i correu brossa, vegeu:

http://www.cert.org/tech_tips/email_bombing_spamming.html.

Enllaç d'interès

Vegeu el cas de

Verisign-Microsoft a:

<http://www.computerworld.com/softwaretopics/os/windows/story/0,10801,59099,00.html>.

formàtica se'n pot anar en orris per una simple trucada o un correu que un usuari malinterpreti.

- **SQL injection:** és una tècnica orientada a bases de dades, i a servidors web en particular, que s'aprofita generalment de programació incorrecta de formularis web, en els quals no es controla correctament la informació que es proporciona: no es determina que la informació d'entrada sigui del tipus correcte (molt tipificada respecte al que s'espera) o no es controla quins tipus o caràcters literals s'introdueixen. La tècnica s'aprofita del fet que els caràcters literals obtinguts pels formularis (per exemple web, tot i que els atacs poden patir-se des de qualsevol API que permeti l'accés a una base de dades, per exemple PHP o PERL) són utilitzats directament per a construir les consultes (en SQL) que atacaran una determinada base de dades (a la qual en principi no es té accés directe). Normalment, si existeixen les vulnerabilitats i hi ha poc control dels formularis, es pot injectar codi SQL al formulari, de manera que es construeixin consultes SQL que proporcionin la informació cercada. En casos dràstics podria obtenir-se la informació de seguretat (usuaris i contrasenyes de la base de dades) i, fins i tot, taules o la base de dades sencera, i també podrien produir-se pèrdues d'informació o esborraments intencionats de dades. En particular, aquesta tècnica en ambients web pot dur a resultats greus (per a l'empresa proveïdora del servei, amb fortes conseqüències econòmiques), a causa de les lleis que protegeixen la privacitat de dades personals, que es poden veure compromeses, fer-se públiques o ser venudes a tercers si s'obtenen per un atac d'aquest tipus. En aquest cas d'atac, més que una qüestió de seguretat del sistema, es tracta d'un problema de programació i control amb tipatge fort de les dades esperades en l'aplicació, a més del control adequat i el coneixement de les vulnerabilitats presents en el programari que es fa servir (base de dades, servidor web, API com PHP, PERL, etc.).
- **Cross-site scripting, XSS:** és una altra problemàtica relacionada amb ambients web, en particular amb alteracions del codi HTML i *scripts* quan un usuari visualitza un determinat lloc web que pot ser alterat dinàmicament. S'aprofita generalment dels errors a l'hora de validar codi HTML (tots els navegadors tenen més o menys problemes, per la mateixa definició de l'HTML original, que permet llegir pràcticament qualsevol codi HTML per incorrecte que sigui). En alguns casos, la utilització de vulnerabilitats pot ser directa, mitjançant *scripts* a la pàgina web, però normalment els navegadors en tenen un bon control. D'altra banda, indirectament hi ha tècniques que permeten inserir codi de *script*, o bé mitjançant accés a les galetes de l'usuari des del navegador, o bé mitjançant l'alteració del procés pel qual es redirecciona una pàgina web a l'altra. També hi ha tècniques mitjançant marcs (*frames*) que permeten redirigir l'HTML que s'està veient o penjar directament el navegador. En particular, poden ser vulnerables els motors de cerca dels llocs web, que poden permetre l'execució de codi de *scripts*. En general, són atacs amb diverses tècniques complexes, però amb l'objectiu de capturar informació com ara galetes, que poden ser usades

en sessions i permetre així la substitució d'una determinada persona mitjançant readreçaments de llocs web o l'obtenció d'informació seva. Novament, des de la perspectiva de sistema, és més una qüestió de programari en ús. Cal controlar i conèixer les vulnerabilitats detectades en navegadors (i aprofitar els recursos que ofereixen per a evitar aquestes tècniques) i controlar l'ús de programari (motors de cerca emprats, versions del servidor web i API utilitzades en els desenvolupaments).

En general, algunes recomanacions (molt bàsiques) per a la seguretat podrien ser:

- Controlar un factor problemàtic: els usuaris. Un dels factors que poden afectar més la seguretat és la confidencialitat de les contrasenyes, i aquesta es veu afectada pel comportament dels usuaris; això facilita a possibles atacants les accions des de l'interior del mateix sistema. La majoria dels atacs solen venir de dins del sistema, és a dir, una vegada l'atacant ja hi ha guanyat l'accés.
- Entre els usuaris, sempre hi ha aquell que és una mica oblidadís (o indiscret), que oblidava la contrasenya cada dos per tres, l'esmenta en converses, l'escriu en un paper que oblidava, o que la deixa escrita al costat (o enganxat) de l'ordinador o sobre la taula de treball, o que simplement la deixa a altres usuaris o coneguts. Un altre tipus és el que col·loca contrasenyes molt previsible, com el seu identificador d'usuari, el seu nom, el seu DNI, el nom de la seva parella, el de la seva mare, el de la seva mascota, etc., coses que amb un mínim d'informació poden trobar-se fàcilment (i més en aquests moments d'apogeu de les xarxes socials, en què és fàcil obtenir aquest tipus d'informació des de diferents xarxes en què participi l'usuari). Un altre cas són els usuaris normals amb un cert coneixement que col·loquen contrasenyes vàlides, però sempre cal tenir en compte que hi ha mecanismes que poden trobar-les (trencament de contrasenyes, detectors, suplantació, etc.). Cal establir una certa **cultura de la seguretat** entre els usuaris i, mitjançant diferents tècniques, obligar-los que canviïn les contrasenyes, que no utilitzin paraules típiques, que usin contrasenyes llargues (amb més de 6 o 8 caràcters, com a mínim), etc. Darrerament, en moltes empreses i institucions s'està implantant la tècnica de fer signar un contracte (o carta de compromisos) a l'usuari, de manera que se l'obliga a no divulgar la contrasenya o cometre actes de vandalisme o atacs des del seu compte (és clar que això no impedeix que d'altres ho facin per ell).
- No utilitzar ni executar programes dels quals no puguem garantir l'origen. Normalment, molts distribuïdors utilitzen mecanismes de comprovació de signatures per a verificar que els paquets de programari són tals, com per exemple les sumes md5 (ordre `md5sum`) o la utilització de signatures GPG [Hatd] (ordre `gpg`). El venedor o distribuïdor proporciona una suma md5 del seu arxiu (o imatge de CD/DVD), de manera que podem comprovar-ne l'autenticitat. Últimament, en les distribucions s'estan fent servir tant

signatures per a paquets individuals com signatures per als dipòsits de paquets, com a mecanisme per a garantir la fiabilitat del proveïdor.

- No utilitzar usuaris privilegiats (com el *root*) per al treball normal de la màquina, ja que qualsevol programa (o aplicació) tindria els permisos per a accedir a qualsevol zona del sistema d'arxius o a serveis en execució.
- No accedir remotament amb usuaris privilegiats ni executar programes que puguin tenir privilegis. I més si no coneixem, o no hem comprovat, els nivells de seguretat del sistema i les seves connexions. En particular, un altre dels problemes actuals són les xarxes sense fil (com Wi-Fi) insegures, ja que, per exemple, alguns xifratges com el que es fa servir en xarxes Wi-Fi WEP són molt febles. No s'han d'usar connexions crítiques sobre xarxes insegures.
- No utilitzar elements (programes o serveis) que no sabem com actuen ni intentar descobrir-ho amb execucions repetides.

Aquestes mesures poden ser poc productives, però si no hem assegurat el sistema, no podem tenir cap control sobre el que pot passar i, tot i així, ningú no garanteix que no es pugui introduir algun programa maliciós que burli la seguretat si l'executem amb els permisos adequats. En general, cal considerar que hem de tenir una vigilància activa amb tot tipus d'activitats que impliquin accessos i execució de tasques de manera més o menys privilegiada i un ús de mecanismes de comunicació insegurs.

1.2. Contramesures

Pel que fa a les mesures que es poden prendre contra els tipus d'atacs presentats, en podem trobar algunes de preventives i de detecció del que succeeix en els nostres sistemes.

Vegem alguns tipus de mesures que podríem prendre en els àmbits de prevenció i detecció d'intrusos (s'esmenten eines útils, algunes de les quals examinarem més endavant):

- **Trencament de contrasenyes (*password cracking*):** en atacs de força bruta per a trencar les contrasenyes sol ser habitual intentar obtenir l'accés per inici de sessió de manera repetida; si s'aconsegueix entrar, la seguretat de l'usuari ha estat compromesa i es deixa la porta oberta a altres tipus d'atacs com, per exemple, les portes secretes o, simplement, a la destrucció del compte. Per a prevenir aquest tipus d'atacs, cal reforçar la política de contrasenyes, demanant una longitud mínima i canvis de contrasenya periòdics. Una cosa que cal evitar és l'ús de paraules comunes en les contrasenyes: molts d'aquests atacs es fan mitjançant la força bruta, amb un fitxer de diccionari (amb paraules en l'idioma de l'usuari, termes comuns,

noms propis, argot, etc.). Aquest tipus de contrasenyes seran les primeres a caure. També pot ser fàcil obtenir informació de l'atacat, com ara noms, DNI o la seva adreça i informació procedent de les seves xarxes socials, i usar aquestes dades per a provar les contrasenyes. Per tot això tampoc no es recomanen contrasenyes amb DNI, noms (propis o de familiars, etc.), adreces, noms de mascotes, etc. Una bona elecció sol ser triar contrasenyes d'entre 6 i 8 caràcters com a mínim i que continguin caràcters alfabètics, numèrics i algun caràcter especial. Malgrat que la contrasenya estigui ben triada, pot ser insegura si s'utilitza en serveis o en xarxes obertes, no segures. Per tant, es recomana reforçar els serveis mitjançant tècniques i serveis de xifratge que protegeixin les contrasenyes i les comunicacions. Al contrari, s'han d'evitar (o no usar) tots aquells serveis que no suportin xifratge i que, consegüentment, siguin susceptibles de ser atacats amb mètodes, per exemple, de detectors; entre aquests podríem incloure serveis com Telnet, FTP, rsh i rlogin, entre d'altres, dels quals ja hi ha alternatives més segures.

- **Explotació de forats:** cal evitar disposar de programes que no s'utilitzin, que siguin antics o que no s'actualitzin (perquè són obsolets). Cal aplicar els últims pedaços i actualitzacions disponibles, tant per a les aplicacions com per al sistema operatiu, provar eines que detectin vulnerabilitats i mantenir-se al dia de les vulnerabilitats que es vagin descobrint. Encara que cal assenyalar que, en determinades ocasions, les pròpies correccions d'alguna fallada de seguretat passada ens en poden portar d'altres de noves. Com a cas particular, cal estar especialment atent a les divulgacions de vulnerabilitats de tipus *Oday*, que acostumen a explotar vulnerabilitats no conegudes i que els desenvolupadors no han tingut temps d'arreglar. En alguns casos, la finestra de vulnerabilitat (temps entre el coneixement de la vulnerabilitat i la seva solució), contra el que podria semblar, podria allargar-se setmanes, o fins i tot anys, i convertir-se en una fallada de seguretat desconeguda durant molt de temps per al públic general, però no així per a cert mercat, que pot involucrar des de *crackers* fins a agències governamentals, les quals aprofiten aquestes *Oday* per a motius polítics (ci-berguerra), invasió de la privadesa o espionatge industrial.
- **Virus:** s'han d'utilitzar mecanismes o programes antivirus, sistemes de filtratge de missatges sospitosos i evitar l'execució de sistemes de macros (que no es puguin verificar). No convé minimitzar els possibles efectes dels virus, cada dia es perfeccionen més i tècnicament és possible crear virus simples que poden desactivar xarxes en qüestió de minuts (només cal observar alguns dels virus dels últims anys en ambients Windows). En especial, si en els nostres sistemes hi ha entorns Windows, seria interessant la possibilitat d'examinar virus que puguin afectar aquests sistemes. Hi ha antivirus per a UNIX i Linux, com ClamAV, que evitaran que es transmetin virus als nostres sistemes interns. En sistemes de correu basats en GNU/Linux, podem establir combinacions de productes antivirus i contra el correu brossa per a evitar aquestes transmissions, com per exemple ClamAV combinat amb altres productes, com SpamAssassin.

Enllaços d'interès

Sobre vulnerabilitats, una bona eina és Nessus. Per a descobrir vulnerabilitats noves, vegeu CERT a: <http://www.cert.org/advisories/>, lloc antic, i <http://www.us-cert.gov/cas/techalerts/index.html>.

Cas Snowden

Vegeu NSA *versus* Snowden: http://en.wikipedia.org/wiki/Edward_Snowden

Enllaços d'interès

Vegeu pedaços i vulnerabilitats per al sistema operatiu a: <http://www.debian.org/security>, <http://www.redhat.com/security>, <http://fedoraproject.org/wiki/Security>.

- **Cucs:** cal controlar l'ús de les nostres màquines o usuaris en hores no previstes, amb patrons de comportament infreqüents, així com el control del trànsit de sortida i entrada.
- **Cavalls de Troia:** s'ha de verificar periòdicament la integritat dels programes mitjançant mecanismes de suma o de signatures, detectar el trànsit anòmal de sortida o entrada al sistema i utilitzar tallafocs per a bloquejar trànsit sospitós. Una versió bastant perillosa dels troians la formen les eines d'intrusió (*rootkits*, que comentarem més endavant), que fan més d'una funció gràcies a un conjunt variat d'eines. Per a la verificació de la integritat, podem utilitzar mecanismes de sumes, com md5 o gpg, o eines que automatitzen aquest procés, com Tripwire o AIDE.
- **Portes secretes:** cal obtenir dels proveïdors o venedors del programari la certificació que no conté cap tipus de porta secreta amagada no documentada i, sens dubte, acceptar el programari provinent només de llocs que ofereixin garanties. Quan el programari sigui de tercers o de fonts que podrien haver modificat el programari original, molts fabricants (o distribuïdors) integren algun tipus de verificació de programari basat en codis de suma o signatures digitals (tipus md5 o gpg) [Hatd]. Sempre que aquestes estiguin disponibles, fóra útil verificar-les abans d'instal·lar el programari. També pot provar-se el sistema de manera intensiva, abans de col·locar-lo com a sistema de producció. Un altre problema pot consistir en l'alteració del programari *a posteriori*. En aquest cas també poden ser útils els sistemes de signatures o sumes per a crear codis sobre programari ja instal·lat i controlar que no es produeixin canvis en programari vital. També resulten útils les còpies de seguretat, amb les quals podem fer comparacions per a detectar canvis.
- **Bombes lògiques:** en aquest cas, s'acostumen a ocultar amb activacions per temps o per accions de l'usuari. Podem verificar que no existeixin en el sistema treballs no interactius introduïts de tipus `crontab`, `at` i altres processos (per exemple, de tipus `nohup`), que disposin d'execució periòdica o que estiguin en execució en segon pla des de fa molt de temps (ordres `w`, `jobs`). En qualsevol cas, podrien utilitzar-se mesures preventives que impedissin treballs programats no interactius als usuaris (`crontab`) o que només els permetessin a aquells que realment els necessitin.
- **Registres de teclat i eines d'intrusió:** en aquest cas hi haurà algun procés intermedi que intentarà capturar les nostres pulsacions de tecles i les emmagatzemarà o comunicarà a algun lloc. Caldrà examinar situacions en què aparegui algun procés estrany pertanyent al nostre usuari, o bé detectar si tenim algun fitxer obert amb el qual no estiguem treballant directament (per exemple, podria ser d'ajuda `lsOf`, vegeu `man`), o bé connexions de xarxa, si es tractés d'un registrador de tecleig amb tramesa externa. Per a provar un funcionament molt bàsic d'un registrador de tecleig molt senzill, pot veure's l'ordre de sistema `script` (vegeu `man script`). L'altre cas,

Enllaç d'interès

L'eina `chkrootkit` es pot trobar a:
<http://www.chkrootkit.org>.

l'eina d'intrusió (que sol incloure també algun registre de teclat) sol ser un paquet d'uns quants programes amb diverses tècniques, i permet a l'atacant, una vegada ha entrat en un compte, utilitzar diversos elements, com un registre de teclat, portes secretes, cavalls de Troia (substituint ordres del sistema), etc., per a obtenir informació i portes d'entrada al sistema. Moltes vegades s'acompanya de programes que netegen els registres, per a eliminar les proves de la intrusió. Un cas particularment perillós el formen les eines d'intrusió (*kernel rootkits*), que s'usen o vénen en forma de mòduls de nucli, la qual cosa els permet actuar en l'àmbit del mateix nucli. Una eina útil per a verificar les eines d'intrusió és `chkrootkit`, o altres alternatives, com ara `rkhunter`.

- **Escaneig de ports:** els escàners solen llançar sobre un o més sistemes, bucles d'escaneig de ports coneguts, per detectar els que queden oberts i aquells en els quals els serveis estaran funcionant (i obtenir, per tant, informació de les versions dels serveis), i així conèixer si podrien ésser susceptibles d'atacs.
- **Detectors:** han d'evitar-se intercepcions i impedir així la possibilitat que s'introdueixin escoltes. Una tècnica és l'ús de maquinari específic per a la construcció de la xarxa, de manera que pugui dividir-se en segments perquè el trànsit només circuli per la zona que s'utilitzarà, posar tallafocs per a unir aquests segments i poder controlar el trànsit d'entrada i sortida. També cal usar tècniques de xifratge perquè els missatges no puguin ser llegits i interpretats per algú que escolti la xarxa. Per al cas tant d'escàners com de detectors, podem utilitzar eines com `Wireshark` (antic `Ethereal`) i `Snort` per a fer comprovacions sobre la nostra xarxa o, per a l'escaneig de ports, `Nmap`. En el cas dels detectors, es poden detectar a la xarxa mitjançant la cerca de màquines en mode Ethernet promiscu (estan a les escoltes de qualsevol paquet que circula). Normalment, la targeta de xarxa només hauria de capturar el trànsit que va cap a ella (o de tipus *broadcast* o *multicast*).
- **Segrest:** en aquest cas algunes contramesures són implementar mecanismes de xifratge en els serveis, demanar autenticació i, si és possible, que aquesta autenticació es renovi periòdicament; controlar el trànsit entrant o sortint mitjançant tallafocs, i monitorar la xarxa per a detectar fluxos de trànsit sospitosos. En casos d'aplicacions web o de correu electrònic, limitar al màxim la utilització de codis de *script* o, senzillament, eliminar la possibilitat d'execució de *scripts* interns de finestres HTML quan provenguin de fonts no fiables.
- **Desbordaments:** solen ser habituals en *bugs* o forats del sistema i solen solucionar-se (si han estat prèviament detectats) mitjançant una actualització del programari o paquet afectat. En qualsevol cas, poden observar-se, gràcies als registres del sistema, situacions estranyes de caiguda de serveis que haurien d'estar funcionant. També poden maximitzar-se els controls de processos i accessos a recursos per a aïllar el problema quan es produeixi

en entorns d'accés controlat, com el que ofereix SELinux. Existeixen altres alternatives, com ara Grsecurity o PaX, que són pedaços dels nuclis oficials de Linux per a obtenir proteccions addicionals en aquest sentit, tant per a problemes de desbordament de *buffers*, com de pila o *heap*.

Vegeu també

SELinux es tracta en l'apartat 4 d'aquest mòdul.

- **Denegació de servei i altres com SYN flood o bombardeig de correu:** s'han de prendre mesures de bloqueig de trànsit innecessari en la nostra xarxa (per exemple, per mitjà de tallafocs). En els serveis en què es pugui, caldrà controlar les mides de la memòria intermèdia, el nombre de clients per atendre, el temps d'espera (*timeouts*) de tancament de connexions, les capacitats del servei, etc.
- **Falsejament d'identitat:** a) falsejament d'IP, b) falsejament d'ARP, c) correu electrònic. Aquests casos necessiten un xifratge fort dels serveis, control per tallafocs i mecanismes d'autenticació basats en diversos aspectes (per exemple, no basar-se en la IP, si pogués veure's compromesa). Es poden aplicar mecanismes que controlin les sessions establertes i que es basin en diversos paràmetres de la màquina alhora (sistema operatiu, processador, IP, adreça Ethernet, etc.). També es poden monitorar sistemes DNS, caus d'ARP, cues de correu, etc., per a detectar canvis en la informació que n'invalidin d'anteriors.
- **Enginyeria social:** no és pròpiament una qüestió informàtica, però també és necessària perquè les persones no empitjorin la seguretat. Existeixen diverses mesures adequades, com augmentar la informació o educar els usuaris (perquè no divulguin dades privades ni informació que pugui oferir pistes sobre les contrasenyes, advertir-los sobre l'ús que facin de les xarxes socials, etc.) i divulgar tècniques bàsiques per a millorar la seva seguretat. També s'han de tenir en compte altres temes de seguretat: controlar quin personal disposarà d'informació crítica de seguretat i en quines condicions pot cedir-la a d'altres. Els serveis d'ajuda i manteniment d'una empresa poden ser un punt crític, i ha de controlar-se qui té informació de seguretat, i com la fa servir.

Respecte als usuaris finals, cal esforçar-se per a millorar la seva cultura de seguretat, tant en contrasenyes (i en la deixadesa del seu ús), com en la gestió del programari que utilitzen diàriament, per tal de mantenir-los actualitzats, així com proporcionar serveis fiables.

2. Seguretat del sistema

Davant dels possibles atacs, hem de disposar de mecanismes de prevenció, detecció i recuperació dels nostres sistemes.

Per a la prevenció local, cal examinar els diferents mecanismes d'autenticació i permisos d'accés als recursos per a poder definir-los correctament, de manera que es garanteixi la confidencialitat i la integritat de la nostra informació.

Fins i tot en aquest cas, no estarem protegits d'atacants que ja hagin obtingut accés al nostre sistema, ni d'usuaris hostils que vulguin saltar-se les restriccions imposades en el sistema.

Respecte a la seguretat en xarxa, hem de garantir que els recursos que oferim (si proporcionem uns determinats serveis) tinguin els paràmetres de confidencialitat necessaris (i encara més si aquests estan garantits per lleis jurídiques nacionals, l'incompliment de les quals pot provocar sancions greus i publicitat negativa indirecta per la falta de compliment). Els serveis oferts no poden ser usats per tercers no desitjats, de manera que un primer pas serà controlar que els serveis oferts siguin els que realment volem i que no estem oferint, a més, altres serveis que no tenim controlats. En el cas de serveis dels quals nosaltres som clients, també caldrà garantir els mecanismes d'autenticació, en el sentit d'accedir als servidors correctes i que no hi hagi casos de suplantació de serveis o servidors (normalment força difícils de detectar a nivell d'usuari).

Quant a les aplicacions i als mateixos serveis, a més de garantir la configuració correcta de nivells d'accés mitjançant permisos i l'autenticació dels usuaris permesos, haurem de vigilar la possible explotació d'errors en el programari. Qualsevol aplicació, per molt ben dissenyada i implementada que estigui, pot tenir un nombre més o menys alt d'errors que poden ser aprofitats per a, amb certes tècniques, saltar-se les restriccions imposades. En aquest cas, practiquem una política de prevenció que passa per mantenir el sistema actualitzat en la mesura del que sigui possible, de manera que, o bé l'actualitzem davant de qualsevol nova correcció, o bé som conservadors i mantenim aquelles versions que siguin més estables en qüestió de seguretat. Normalment, això significa verificar periòdicament uns quants llocs de seguretat, de solvència coneguda, per a conèixer les últimes fallades detectades en el programari, tant d'aplicacions com de sistema, i les vulnerabilitats que se'n deriven i que podrien exposar els nostres sistemes a fallades de seguretat, localment o per xarxa.

Xifratge de sistemes de fitxers

En els sistemes GNU/Linux, s'implementen diverses opcions per al xifratge de dades, per exemple destacar *Encfs*, un sistema de fitxers en espai d'usuari que ens permet xifrar les dades d'usuari; vegeu: <http://www.arg0.net/encfs>

3. Seguretat local

La seguretat local [Pen, Hatb] és bàsica per a la protecció del sistema [Deb, Hatc], ja que, normalment, després d'un primer intent d'accés des de la xarxa, és la segona barrera de protecció abans que un atac aconseguixi fer-se amb part del control de la màquina. A més, la majoria dels atacs acaben fent ús de recursos interns del sistema.

Accés local

Diversos atacs, tot i que procedeixin de l'exterior, tenen com a finalitat aconseguir un accés local.

3.1. Bootloaders

Respecte a la seguretat local, ja en l'arrencada se'ns poden presentar problemes a causa de l'accés físic que un intrús pogués tenir a la màquina.

Un dels problemes es troba en la mateixa arrencada del sistema. Si el sistema pot arrencar-se des de dispositius extraïbles o des de CD/DVD, un atacant podria accedir a les dades d'una partició GNU/Linux (o també en entorns Windows) solament muntant el sistema de fitxers i podria col·locar-se com a usuari *root* sense necessitat de conèixer cap contrasenya. En aquest cas, cal protegir l'arrencada del sistema des de la BIOS, per exemple, protegint-ne l'accés mitjançant una contrasenya, de manera que no es permeti l'arrencada des de CD/DVD (amb un CD autònom o *live CD*, per exemple), USB o altres connexions externes.

També és raonable actualitzar la BIOS, ja que també pot tenir errors de seguretat. A més, cal anar amb compte, perquè la majoria de fabricants de BIOS ofereixen contrasenyes addicionals conegudes (una espècie de porta secreta), de manera que no podem dependre d'aquestes mesures en exclusiva.

El pas següent és protegir el carregador de l'arrencada (*bootloader*), Lilo o Grub, de manera que l'atacant no pugui modificar les opcions d'arrencada del nucli o modificar directament l'arrencada (cas de Grub). Qualsevol dels dos pot protegir-se també per mitjà de contrasenyes addicionals.

En Grub-Legacy (Grub1), el fitxer `/sbin/grub-md5-crypt` demana la contrasenya i genera una suma md5 associada. En algunes distribucions en què no està disponible aquesta ordre, es pot fer de manera alternativa durant l'arrencada: en carregar Grub, accedim amb la tecla `c` a *shell* de Grub en el moment d'inici i introduïm `md5crypt` per obtenir el *hash* md5 associat a una contrasenya que proposem.

En Grub, el fitxer `/sbin/grub-md5-crypt` demana la contrasenya i genera una suma md5 associada. En algunes distribucions en què no està disponible aquesta ordre, pot fer-se de manera alternativa durant l'arrencada: en carregar Grub, accedim amb la tecla `C` a l'interpret d'ordres de Grub i hi introduïm `md5crypt` per a obtenir el *hash* md5 associat a una contrasenya que proposem.

Després, el valor obtingut s'introdueix a `/boot/grub/grub.conf`. Sota la línia `timeout` s'introdueix:

```
password --md5 suma-md5-calculada
```

Per a Lilo es col·loca bé una contrasenya global amb

```
password=contrasenya
```

o bé una en la partició que vulguem:

```
image = /boot/vmlinuz-version
password = contrasenya
restricted
```

En aquest cas `restricted` indica, a més, que no es podran canviar els paràmetres que es passen al nucli des de l'indicador d'ordres. Cal anar amb compte de posar el fitxer `/etc/lilo.conf` protegit en el mode de només lectura/escriptura des del *root* (`chmod 600`), si no, qualsevol podria llegir les contrasenyes.

En el cas de Grub 2, és possible fer-ho mitjançant *menuentry*, disponible en la configuració, definint *passwords* per a usuaris concrets, que després es poden afegir a cada *menuentry* definida, a més hi ha la possibilitat d'especificar un superusuari que disposa d'accés a totes les entrades. Un exemple simple de configuració seria el següent: un superusuari que té accés a totes les entrades, mentre que un segon usuari només pot accedir a la segona entrada.

```
set superusers=rootgrub
password rootgrub password1
password otheruser password2
```

```
menuentry "GNU/Linux" {
set root=(hd0,1)
linux /vmlinuz
}
```



```
menuentry "Windows" --users otheruser {  
  set root=(hd0,2)  
  chainloader +1  
}
```

En aquest cas, la configuració de Grub2 fa servir *passwords* en net, i cal assegurar-se que el fitxer no disposi d'accés de lectura per a altres usuaris, o en el seu lloc fer servir mètodes alternatius per a la creació de *passwords*, per exemple, mitjançant *grub-mkpasswd-pbkdf2*, que ens permetrà generar *hashes* del *password* que cal utilitzar. En cas de voler deixar *menuentries* disponibles per a tots, també pot usar-se *-unrestricted* en la definició de l'entrada.

En aquest darrer punt val la pena recordar una recomanació general: els fitxers amb informació sensible mai no haurien d'estar disponibles amb permisos de lectura a tercers, i sempre que sigui possible hauríem d'evitar-los o xifrar-los. En particular, per a dades sensibles, seria desitjable algun tipus de sistema de fitxers que permetés el xifratge dels fitxers presents (o fins i tot el disc o sistema sencer). En aquest sentit podem destacar Encryptfs i EncFS, que ja està disponible en diverses distribucions, per al xifratge dels arxius dels directoris privats dels usuaris o el sistema de fitxers sencer.

Un altre tema relacionat amb l'arrencada és la possibilitat que algú que tingui accés al teclat reiniciï el sistema, ja que si es prem CTRL + ALT + DEL (en diverses distribucions aquesta opció ja sol estar desactivada per defecte), es provoca una operació de tancament en la màquina. Aquest comportament és definit (en un sistema amb sysvinit) a */etc/inittab*, amb una línia com ara:

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

Si es posa com a comentari, aquesta possibilitat de reiniciar quedarà desactivada. O, al contrari, pot crear-se un fitxer */etc/shutdown.allow*, que permet que usuaris determinats puguin apagar i/o reiniciar.

3.2. Contrasenyes i *shadow*s

Les contrasenyes típiques dels sistema UNIX inicials (i de les primeres versions de GNU/Linux) estaven xifrades mitjançant uns algorismes DES (però amb claus petites, i una crida de sistema s'encarregava de xifrar i desxifrar, en concret *crypt*, vegeu-ne el man).

Normalment, es trobaven al fitxer */etc/passwd*, en el segon camp; així per exemple:

```
user:sndb565sadsd:...
```

Sistemes de fitxers xifrats

Els sistemes de fitxers xifrats també són una bona opció per a protegir les dades sensibles d'usuaris, com per exemple, entre d'altres, Ecryptfs i EncFS. Vegeu, per exemple, Encfs: <http://www.arg0.net/encfs>
Una altra opció són sistemes de xifratges de disc, com dm-crypt/LUKS: <https://code.google.com/p/cryptsetup/wiki/DMCrypt>

Però el problema resideix en el fet que aquest fitxer era llegible per qualsevol usuari del sistema, de manera que un atacant podia obtenir el fitxer i utilitzar un atac de força bruta, fins a desxifrar les contrasenyes contingudes en el fitxer, o bé, amb atacs de força bruta una mica més intel·ligents, per mitjà de diccionaris.

El primer pas és utilitzar els fitxers `/etc/shadow`, on es desa ara la contrasenya. Aquest fitxer només és llegible per l'usuari `root`. En aquest cas, a `/etc/passwd` apareix un asterisc (*) on abans era la contrasenya xifrada. Per defecte, les distribucions de GNU/Linux actuals usen contrasenyes de tipus *shadow*, tret que se'ls digui que no les usin.

Un segon pas és canviar el sistema de xifratge de les contrasenyes per un de més complex i difícil de trencar. Ara, tant Fedora com Debian ofereixen contrasenyes per a md5; normalment ens deixen escollir el sistema en temps d'instal·lació. Cal anar amb compte amb les contrasenyes md5, ja que si fem servir NIS podríem tenir algun problema, segons les versions; si no, tots els clients i servidors usaran md5 per a les contrasenyes. Les contrasenyes es poden reconèixer a `/etc/shadow` perquè vénen amb un prefix "`id`" amb `id=1`. No són els únics algorismes de xifratge que poden usar-se, `id=5` és sha-256 o `id=6` és sha-512. De fet, per a utilitzar contrasenyes xifrades més fiables, es pot fer servir (en algunes distribucions) l'ordre `mkpasswd`, a la qual pot aportar-se el xifratge utilitzat i la contrasenya que s'ha de xifrar (vegeu els man de `mkpasswd` i `crypt`).

Altres possibles actuacions consisteixen a obligar els usuaris a canviar la contrasenya amb freqüència (l'ordre `change` pot ser útil), imposar restriccions en la mida i el contingut de les contrasenyes i validar-les amb diccionaris de termes comuns (definint aquestes polítiques per una combinació de paràmetres en `/etc/passwd` o mitjançant control per mòduls PAM).

Pel que fa a les eines, és interessant disposar d'un trencador de contrasenyes (és a dir, un programa per a provar i trencar contrasenyes) per a comprovar la situació real de seguretat dels comptes dels nostres usuaris i forçar així el canvi en les que detectem insegures. Dues de les més utilitzades pels administradors són John the Ripper (paquet `john`) i `crack`. Poden funcionar també per diccionari, de manera que és interessant disposar d'algun ASCII d'espanyol o català (poden trobar-se a la Xarxa).

Una qüestió que s'ha de tenir en compte sempre és fer aquestes proves sobre els nostres sistemes. No s'ha d'oblidar que els administradors d'altres sistemes (o el proveïdor d'accés o ISP) tindran habilitats sistemes de detecció d'intrusos i podem ser objecte de denúncia per intents d'intrusió davant de les autoritats competents (unitats de delictes informàtics) o en el nostre ISP perquè se'ns tanqui l'accés. Cal anar amb molt de compte amb l'ús d'eines de seguretat, que són sempre a la frontera entre ser de seguretat o d'intrusió.

3.3. Bits *sticky* i *setuid*

Un altre problema important són alguns permisos especials que són utilitzats sobre fitxers o *scripts*.

El bit *sticky* s'utilitza sobretot en directoris temporals, on volem que, en alguns grups (de vegades no relacionats), qualsevol usuari pugui escriure, però només pugui esborrar el propietari del directori o bé el propietari del fitxer que estigui en el directori. Un exemple clàssic d'aquest bit és el directori temporal `/tmp`. Cal vigilar que no hi hagi directoris d'aquest tipus, ja que poden permetre que qualsevol hi escrigui, per la qual cosa caldrà comprovar que no n'hi hagi més que els estrictament necessaris com a temporals. El bit es col·loca mitjançant (`chmod +t dir`) i pot treure's amb `-t`. En un `ls` apareixerà com un directori amb permisos `drwxrwxrwt` (observeu l'última `t`).

El bit *setuid* permet a un usuari executar (o bé un programa binari executable o bé un *script* de l'interpret d'ordres) amb els permisos d'un altre usuari. Això en algun cas pot ser d'utilitat, però és potencialment perillós. És el cas, per exemple, de programes amb *setuid* de *root*: un usuari, malgrat que no tingui permisos de *root*, pot executar un programa amb *setuid* que pot disposar de permisos interns d'usuari *root*. Això és molt perillós en cas de *scripts*, ja que podrien editar-se i modificar-se per a fer qualsevol tasca. Per tant, cal tenir controlats aquests programes i, en cas que no es necessiti el *setuid*, eliminar-lo. El bit es col·loca mitjançant `chmod +s`, aplicant-lo o bé al propietari (llavors s'anomena *suid*) o bé al grup (s'anomena bit *sgid*); pot treure's amb `-s`. En el cas de visualitzar amb `ls`, el fitxer apareixerà amb `-rwsrwsrws` (observeu la `S`) si és només *suid*, en *sgid* la `S` apareixeria després de la segona `w`.

En cas d'utilitzar `chmod` amb notació octal, es fan servir quatre xifres, en què les tres últimes són els permisos clàssics `rwXrwxrwx` (recordeu que cal sumar en la xifra 4 per a `r`, 2 `w` i 1 per a `x`) i la primera té un valor per a cada permís especial que es vulgui (que se sumen): 4 (per a *suid*), 2 (*sgid*) i 1 (per a *sticky*).

Fóra desitjable fer una anàlisi periòdica del sistema de fitxers per a detectar els fitxers amb *suid* i *sgid* en el sistema i determinar si són realment necessaris o poden provocar problemes de seguretat.

3.4. Habilitació de *hosts*

En el sistema hi ha uns quants fitxers de configuració especials que permeten habilitar l'accés a una sèrie d'amfitrions (*hosts*) per a alguns serveis de xarxa, però els errors dels quals poden permetre atacar després la seguretat local. Ens podem trobar amb:

- `.rhosts` d'usuari: permet que un usuari pugui especificar una sèrie de màquines (i usuaris) que poden usar el seu compte mitjançant ordres "`r`" (`rsh`, `rcp`, etc.) sense necessitat d'introduir la contrasenya del compte. Això és

potencialment perillós, ja que una mala configuració de l'usuari podria permetre entrar a usuaris no desitjats, o que un atacant (amb accés al compte de l'usuari) canviés les adreces a `.rhosts` per poder entrar còmodament sense cap control. Normalment, no s'hauria de permetre crear aquests arxius i, fins i tot, caldria esborrar-los completament i deshabilitar les ordres "r". Una anàlisi periòdica del sistema serà útil per a detectar la creació d'aquests fitxers.

- `/etc/hosts.equiv`: és exactament el mateix que els fitxers `.rhosts` però a nivell de màquina, i especifica quins serveis, quins usuaris i quins grups poden accedir sense control de contrasenya als serveis "r". A més, un error com posar en una línia d'aquest fitxer un "+" permet l'accés a "qualsevol" màquina. Normalment, avui en dia tampoc no sol existir per defecte aquest fitxer creat, i sempre existeixen com a alternativa als "r" els serveis de tipus ssh.
- `/etc/hosts.lpd`: en el sistema d'impressió LPD s'utilitzava per a habilitar les màquines que podien accedir al sistema d'impressió. Cal tenir especial cura, si no estem servint impressió, de deshabilitar completament l'accés al sistema, i en cas que ho estiguem fent, restringir al màxim les màquines que realment en fan ús. També es pot intentar canviar a un sistema d'impressió CUPS, per exemple, que té molt més control sobre els serveis. El sistema d'impressió LPD havia estat un blanc habitual d'atacs de tipus cuc o de desbordament, i estan documentats diversos errors importants. Cal estar atents si encara utilitzem aquest sistema i el fitxer `hosts.lpd`.

3.5. Mòduls PAM

Els mòduls PAM [Pen, Mor03] són un mètode que permet a l'administrador controlar com es fa el procés d'autenticació dels usuaris per a determinades aplicacions. Les aplicacions han d'haver estat creades i enllaçades a les biblioteques PAM. Bàsicament, els mòduls PAM són un conjunt de biblioteques compartides que poden incorporar-se a les aplicacions com a mètode per a controlar l'autenticació dels usuaris. A més a més, es pot canviar el mètode d'autenticació (mitjançant la configuració dels mòduls PAM) sense que calgui canviar l'aplicació.

Els mòduls PAM (les biblioteques) acostumen a estar a `/lib/security` o `/lib64/security` (en forma de fitxers objecte carregables dinàmicament). La configuració de PAM és en el directori `/etc/pam.d`, on apareix un fitxer de configuració de PAM per a cada aplicació que està usant mòduls PAM. Ens trobem amb la configuració d'autenticació d'aplicacions i serveis com ssh, d'inici de sessió de l'entorn gràfic d'X Window System, com *xdm*, *gdm*, *kdm*, *xscreensaver*, etc. o, per exemple, de l'inici de sessió del sistema (l'entrada per a identificador d'usuari i contrasenya). En les antigues versions de PAM, s'utilitzava un arxiu de configuració general (típicament a `/etc/pam.conf`), que era on es llegia la configuració PAM si el directori `/etc/pam.d` no existia.

Enllaç d'interès

Si voleu saber més coses sobre els mòduls PAM, podeu consultar la *The Linux-PAM System Administrators Guide* a <http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html>.

La línia típica d'aquests fitxers (a `/etc/pam.d`) tindria aquest format (si s'utilitzava l'antic `/etc/pam.conf` caldria afegir el servei al qual pertany com a primer camp):

```
module-type control-flag module-path arguments
```

on s'especifica:

- 1) Tipus de mòdul: si és un mòdul que requereix que l'usuari s'autentifiqui (`auth`) o és d'accés restringit (`account`); coses que cal fer quan l'usuari entra o surt (`session`), o s'ha d'actualitzar la contrasenya (`password`).
- 2) *Flags* de control: especifiquen si és necessari (`required`), si és un requisit previ (`requisite`), si és suficient (`sufficient`) o si és opcional (`optional`). Aquesta és una de les possibles sintaxis, però per a aquest camp existeix una alternativa més actual que treballa en parelles valor i acció.
- 3) El camí (*path*) del mòdul.
- 4) Arguments que es passen al mòdul (depenen de cada mòdul).

A causa del fet que alguns serveis necessiten diverses línies de configuració comunes, hi ha la possibilitat d'operacions d'inclusió de definicions comunes d'altres serveis. Per a això només cal afegir una línia amb:

```
@include servei
```

Un petit exemple de l'ús de mòduls PAM (en una distribució Debian), pot ser el seu ús en el procés d'inici de sessió (*login*) (també hi ha la llista de les línies incloses provinents d'altres serveis):

```
auth      requisite pam_securetty.so
auth      requisite pam_nologin.so
auth      required  pam_env.so
auth      required  pam_unix.so nullok
account   required  pam_unix.so
session   required  pam_unix.so
session   optional  pam_lastlog.so
session   optional  pam_motd.so
session   optional  pam_mail.so standard noenv
password  required  pam_unix.so nullok obscure min = 4 max = 8 md5
```

Això especifica els mòduls PAM necessaris per a controlar l'autenticació d'usuaris en l'inici de sessió. Un dels mòduls `pam_unix.so` és el que realment fa la verificació de la contrasenya de l'usuari (examina com a dades fitxers `passwd`, `shadow`, etc.).

Altres mòduls controlen la sessió per veure quan ha entrat per darrera vegada, o desen quan entra i surt (per a l'ordre `lastlog`). També hi ha un mòdul que

s'encarrega de verificar si l'usuari té correu per llegir (també cal autenticar-se) i un altre que controla que es canviï la contrasenya (si està obligat a fer-ho en el primer inici de sessió que faci) i que tingui de 4 a 8 lletres. Pot utilitzar-se md5 per al xifratge de contrasenyes.

En aquest exemple podríem millorar la seguretat dels usuaris: el `auth` i el `password` permeten contrasenyes de longitud nul·la: és l'argument `nullok` del mòdul. Això permetria tenir usuaris amb contrasenyes buides (font de possibles atacs). Si traiem aquest argument, ja no permetem contrasenyes buides en el procés d'inici de sessió. El mateix pot fer-se en el fitxer de configuració de `passwd` (en aquest cas, l'ordre de canvi del contrasenya), que també presenta el `nullok`. Una altra possible acció és incrementar en ambdós fitxers la mida màxima de les contrasenyes, per exemple, amb `max = valor`.

3.6. Alteracions del sistema

Un altre problema pot ser l'alteració d'ordres o configuracions bàsiques del sistema, mitjançant la introducció de troians o portes secretes en el sistema, per la simple introducció de programari que substitueixi o modifiqui lleugerament el comportament del programari de sistema.

Un cas típic és la possibilitat de forçar l'usuari `root` perquè executi ordres falses de sistema; per exemple, si el `root` inclogués el `."` en la variable de `PATH`, això permetria l'execució d'ordres des del seu directori actual, cosa que habilitaria la col·locació d'arxius que substituïssin ordres del sistema i que serien executades en primer terme, abans que les de sistema. El mateix procés pot fer-se amb un usuari, tot i que com que els permisos que té són més limitats, pot no afectar tant el sistema, sinó més aviat la seguretat de l'usuari. Un altre cas típic és el de les pantalles d'inici de sessió falses, que poden substituir el típic procés d'inici de `login`, `passwd`, per un programa fals que emmagatzemi les contrasenyes introduïdes.

En cas d'aquestes alteracions, serà imprescindible usar polítiques d'auditoria de canvis del sistema, o bé per mitjà de càlcul de signatures o sumes (gpg o md5), o bé mitjançant algun programari de control com `Tripwire` o `AIDE`. Per als troians podem fer diferents tipus de deteccions, o utilitzar eines com `chkrootkit` o `rkhunter`, si aquestes vinguessin de la instal·lació d'alguna eina d'intrusió coneguda.

3.7. Recursos limitats, *cgroups* i *chroot*

La gestió comuna dels processos existents de la màquina, tant si són perfectament vàlids com nocius (intencionadament o per distracció), ens pot provocar situacions d'esgotament dels recursos, en forma de CPU disponible, memòria disponible, recursos de xarxa o simplement sessions d'usuari concurrents i/o processos executant-se.

Enllaç d'interès

Teniu més informació sobre AusCert UNIX checklist a <http://www.auscert.org.au/5816>.

Enllaç d'interès

Teniu més informació sobre `chkrootkit` a <http://www.chkrootkit.org>.

Per a controlar aquest problema d'esgotament, podem introduir límits a alguns dels recursos usats mitjançant l'ordre `ulimit`, encara que la configuració global de límits es manté en el fitxer `/etc/security/limits.conf`. De fet, aquests límits, que poden ser imposats sobre temps de CPU, nombre màxim de processos, quantitat de memòria i altres de similars, són llegits per mòduls PAM durant el procés de *login* dels usuaris, i establerts per defecte a partir dels límits imposats a cada usuari.

L'elecció dels límits també es definirà pel perfil de funcionament del sistema: tenim un gran nombre d'usuaris al qual hem de limitar els recursos, algun servidor costós en memòria, o per contra només necessitem uns pocs processos per a mantenir un servidor simple. Si podem perfilar la tipologia del nostre servidor, podrem imposar límits raonables que ens permetin posar límit a situacions problemàtiques; pot succeir fins i tot que aquestes s'estiguin donant per atacs de seguretat orientats al consum de recursos per a deixar inoperants els servidors o el sistema en conjunt. Normalment ens concentrarem a controlar, i limitar adequadament, els recursos emprats pel *root* i pels usuaris comuns per a intentar evitar possibles atacs o comportaments defectuosos d'aplicacions corrompudes en el seu funcionament i l'ús dels recursos.

No establir límits i usar el comportament per defecte pot fer caure el nostre sistema, amb certa facilitat, en processos en mal funcionament (per mala programació o descuits de sintaxi), tant si són executables de sistema com si són *scripts*; per exemple, és bastant fàcil crear el que es denomina una *Fork Bomb*. En sistemes UNIX (Linux inclòs), la crida a sistema *fork()* és la que permet crear un procés derivat (fill) a partir de la imatge d'un procés inicial (procés pare). Posteriorment, el procés fill pot especialitzar el seu codi mitjançant altres crides a sistema i/o comunicar-se amb el procés pare. De fet, aquesta és la informació de parentiu que s'observa entre els processos en *ps* amb els seus PID i PPID (Parent PID). Si aquest tipus de crides entren en bucle infinit (més aviat finit, fins que s'esgoten els recursos), ja sigui per distracció, mala programació o intencionadament, la qual cosa aconseguirem en una màquina sense control dels límits, bàsicament farem caure el sistema directament (o el mantindrem en uns extrems de servei mínims).

En el cas de `/etc/security/limits.conf`, cada línia permet uns camps denominats:

```
domini tipus item valor
```

on el *domini* és el *login* d'un usuari o grup. El tipus serà *soft* o *hard* en funció de si és un límit amb certa laxitud o per contra no es pot sobrepassar. L'ítem és referit al tipus de límit (*fsize*, *nofile*, *cpu*, *nproc*, *maxlogins*), que es refereixen a: grandària màxima de fitxer, nombre de fitxers oberts, màxima CPU en minuts usada, nombre total de processos de l'usuari i nombre màxim de *logins* de l'usuari, i finalment el *valor* corresponent estable. Normalment el més ra-

onable és restringir als usuaris el nombre màxim de processos (per a limitar problemes com *fork bomb*) i, tret que hi hagi alguna raó explícita, podem deixar sense ús altres límits. En alguns casos hi ha alternatives. Per exemple, en el cas de l'ús de disc, els sistemes de fitxers podrien aportar control de quotes d'espai de disc per usuari.

Si, per exemple, volguéssim limitar a 64 els processos màxims per usuari, l'entrada de `limits.conf` seria:

```
* hard nproc 64
```

Podem comprovar el funcionament sortint i tornant a entrar en el compte d'usuari per a observar la sortida de l'ordre `ulimit -a`.

Els límits per defecte en la majoria de distribucions acostumen a ser bastant folgats. En una Debian, per exemple, el cas per defecte (i el valor anterior proposat, 64, contra el qual exhibeix per defecte la distribució, 16.006 processos màxims per usuari) seria:

```
# ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e) 0
file size               (blocks, -f) unlimited
pending signals         (-i) 16006
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files              (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority      (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes      (-u) 16006
virtual memory          (kbytes, -v) unlimited
file locks              (-x) unlimited
```

Amb una correcta posada en escena de límits raonables segons el perfil del nostre sistema i dels nostres usuaris, podem mitigar de manera molt important els perills de caiguda dels nostres servidors.

En particular, per a l'establiment de límits, cal assenyalar que també disposem d'una funcionalitat en un nivell de kernel, introduïda en les últimes branques i feta servir majoritàriament pel sistema *Systemd*, denominada *cgroups*, que permet controlar i sol·licitar recursos de sistema de diferents tipus: CPU,

memòria o accés a la xarxa. Podem examinar la llista dels subsistemes suportats amb `cat /proc/cgroups`. S'aporta també una llibreria (*libcgroup*) i un conjunt d'utilitats administratives (*libcgroup-tools*) que poden ser usades per a controlar aquests grups, i monitorar l'ús dels recursos i els límits imposats. *Cgroups* s'utilitza a partir d'un pseudosistema de fitxers, *cgroupfs*, en el qual se situen els recursos en grups de control, als quals es poden associar processos que els estiguin usant i limitar el seu ús. Per exemple, podem esbrinar quins grups trobem i on estan muntats, amb tipus de *filesystem cgroupfs*, on podem observar els directoris i la informació de control de cada grup de recursos mantinguda. També podem observar la configuració en `/etc/cgconfig.conf` (encara que algunes d'aquestes les durà a terme *Systemd* en temps d'arrencada):

```
# cat /proc/cgroups
#subsys_name hierarchy num_cgroups enabled
cpuset 2 1 1
cpu 3 1 1
cpuacct 3 1 1
memory 4 1 1
devices 5 1 1
freezer 6 1 1
net_cls 7 1 1
blkio 8 1 1
perf_event 9 1 1
net_prio 7 1 1
hugetlb 10 1 1

# lsusbsys -am
cpuset /sys/fs/cgroup/cpuset
cpu,cpuacct /sys/fs/cgroup/cpu,cpuacct
memory /sys/fs/cgroup/memory
devices /sys/fs/cgroup/devices
freezer /sys/fs/cgroup/freezer
net_cls,net_prio /sys/fs/cgroup/net_cls,net_prio
blkio /sys/fs/cgroup/blkio
perf_event /sys/fs/cgroup/perf_event
hugetlb /sys/fs/cgroup/hugetlb
```

Posem un exemple de control per a una aplicació costosa de recursos que podríem controlar amb *cgroups* especificada en `/etc/cgconfig.conf`. Segui APP l'executable de l'aplicació i \$USER l'usuari que l'executarà:

```
# Prevenir que APP obtingui tota la memòria
# i CPU disponibles
group APP {
    perm {
```

```
    admin {
        uid = $USER;
    }
    task {
        uid = $USER;
    }
}

# 4 cores usados
#(suposem un sistema amb més disponibles)
cpuset {
    cpuset.mems="0";
    cpuset.cpus="0-3";
}

memory {
# 4 GiB limit
    memory.limit_in_bytes = 4294967296;
}
}
```

Aquest *cgroup* permetrà a APP usar els *cores* 0 a 3 (4 en total) i limitar el seu ús de memòria a 4 GB, i la posarem en execució amb el límit de recursos amb (no s'ha d'oblidar substituir APP i \$USER pel nom de l'aplicació i usuari):

```
$ cgexec -g memory,cpuset:APP /opt/APPdir/APP -opcions
```

Respecte als *cgroups*, recentment, en la versió de kernel 3.16 (agost de 2014), s'ha establert un nou mètode d'accés que permet disposar-los com una única jerarquia.

Examinem una tercera possibilitat per al control de límits de recursos lliurats a una determinada aplicació, servei o usuari. És el que es coneix com a *creació d'ambients chroot*, el concepte dels quals és la creació d'una espècie d'ambient de presó (conegut com a *jail*), en què posem uns límits molt clars a allò que es pot fer sense afectar la resta del sistema.

Es basa en una crida al sistema *chroot()* que bàsicament redefineix el que el procés fill associat percep com a directori arrel (/), que, per a l'ambient (o gàbia) creat, pot associar-se en una determinada posició del sistema de fitxers, percebuda pel procés com a arrel (/), i buscarà a partir d'allà totes les configuracions, components del sistema, biblioteques, etc. des d'aquesta falsa arrel.

Ja que es canvia l'estructura del sistema de fitxers per una de nova, el procés només funcionarà si disposa de totes les seves dades a la zona nova. Per tant, haurem de replicar totes les dades, fitxers i ordres necessàries.

cgroups info

Podeu consultar la documentació del kernel a <https://www.kernel.org/doc/Documentation/cgroups/>

Enllaç d'interès

Novetats *cgroups* en kernel 3.16: <http://lwn.net/Articles/601840/>

Convé recalcar que un ambient de *chroot* protegeix contra l'accés a fitxers fora de la zona (de gàbia), però no protegeix contra altres límits, com la utilització del sistema, accessos a memòria o d'altres de similars.

Una vegada escollim el procés que volem restringir, podem copiar els seus fitxers de dades en l'estructura adequada de directoris i podem examinar els executables del procés amb *ldd executable*, la qual cosa ens aporta informació de les biblioteques necessàries per a la seva execució, que poden ser copiades i situades en els punts de l'estructura de fitxers adequats.

Un cop disposem de l'estructura simplement amb:

```
chroot /nova/arrel executable
```

li passem l'execució a la seva nova gàbia (dins de l'arrel obtinguda resolent el camí */nova/arrel*).

3.8. Protecció d'executables mitjançant Hardening-Wrapper

Amb aquest nom es coneixen una sèrie de tècniques usades per a mitigar diversos problemes de seguretat relacionats amb atacs duts a terme per executables. En concret, es tracta dels problemes de desbordament de *stack* (pila), *heap* i proteccions contra l'accés a zones de memòria de dades i codi dels executables.

S'utilitzen aquestes tècniques en associació amb el compilador (per exemple, GNU *gcc*), mitjançant una sèrie de paràmetres i opcions (*flags*) passades en temps de compilació, ja sigui per a la compilació d'aplicacions d'usuari, clients de serveis o la part *server* de serveis (*daemons*), que a partir de les seves fonts de codi i el procés de compilació són protegides contra diverses tècniques d'atac als executables.

En Debian podem instal·lar les utilitats i fitxers complementaris per al procés de compilació, a més de disposar de la utilitat *hardening-check*, que ens permet comprovar les proteccions de què disposa un determinada ordre, aplicació o *daemon*. Posem l'exemple de la instal·lació dels paquets en Debian i la comprovació d'un *daemon* (*sshd* en aquest cas):

```
# apt-get install hardening-wrapper hardening-includes

# hardening-check /usr/sbin/sshd
/usr/sbin/sshd:
  Position Independent Executable: yes
  Stack protected: yes
```

```
Fortify Source functions: yes (some protected functions found)
Read-only relocations: yes
Immediate binding: yes
```

Entre les comprovacions que s'efectuen en l'executable (i les relacions amb paràmetres *[flags]* de compilació en GNU gcc, els que s'esmenten poden ser passats per línia d'ordres a gcc, o bé en el corresponent fitxer de Makefile que efectui el procés de compilació):

- *Position Independent Executable*, que permet al programa que la seva posició en memòria del sistema sigui movable, sense disposar necessàriament de posició fixa i fins i tot de manera aleatòria entre diferents execucions. En gcc aconseguim aquesta funcionalitat amb els *flags* de compilació *-pie* aplicats a tots els components de l'executable. El kernel que maneja els processos ha de tenir suport d'allò que es coneix com ASLR (*Address Space Layout Randomization*), que permet protecció contra atacs de desbordament de *buffer* (*buffer overflows*). Per a prevenir que un atacant pugui atacar una determinada funció en memòria, ASLR reordena de manera aleatòria àrees clau del procés (la imatge de l'executable en memòria) per a canviar posicions de la base de la part executable, així com les posicions de l'*stack*, el *heap* o les biblioteques dinàmiques dins de l'espai d'adreces del procés. Així s'aconsegueix que sigui difícil predir les posicions de memòria quan s'intenten atacs amb aquesta fi. Pot esbrinar-se si hi ha suport en el kernel per a ASLR examinant */proc/sys/kernel/randomize_va_space*, que hauria de donar un valor d'1 o 2 per a opcions amb ASLR activades i 0 quan estan desactivades.
- *Stack Protected*. En aquest cas s'han ofert mètodes addicionals per a protegir els desbordaments de *stack*, per exemple, mitjançant la compilació gcc amb l'opció (*-fstack-protector*).
- *Fortify Source functions*. En aquest cas, quan durant el procés de compilació (en la part de *link*) els executables són enllaçats amb les funcions de la biblioteca estàndard C (funcions de *glibc* en un sistema GNU/Linux), hi ha una certa substitució d'algunes funcions per unes altres. Algunes funcions de *glibc* són conegudes per no ser segures en diversos aspectes, encara que tenen alternatives segures en la mateixa biblioteca, ja que tenen tests de seguretat incorporats, per exemple, enfront de *buffer overflows*. En el cas de gcc, en el procés de compilació es fa amb *-D_FORTIFY_SOURCE=2 -O1* (o superior nivell d'optimització).
- *Read-only relocations*. Això permet al *linker*, quan detecta zones de memòria que pot deduir que seran de sol accés de lectura durant el procés de compilació, marcar-les com a tals abans de començar l'execució. D'aquesta manera, es redueix la possibilitat que un atacant pugui usar aquestes àrees per a injectar codi o que aprofiti *exploits* de corrupció de memòria. En compilació, poden passar-se a gcc els *flags -Wl,-z,relro* per a aquesta opció.

- *Immediate binding*. És una possible combinació amb l'anterior, que permet al *linker*, prèviament a l'execució, resoldre moviments de dades i certs enllaços a memòria que evitin que aquests moviments (*relocates*) es portin a terme *a posteriori*. Així, en combinació amb l'opció anterior, es redueixen de manera sensible les zones de memòria disponibles contra atacs de corrupció de memòria.

En el cas de gcc, la majoria d'aquestes tècniques poden activar-se per defecte abans de la compilació activant la variable següent (amb valor igual a 1 per a activar o 0 per a desactivar):

```
$ export DEB_BUILD_HARDENING=1
$ gcc ....
```

Aquestes tècniques aporten una bona base per a protegir els executables del sistema contra atacs a la seva forma de procés mentre estan en execució, així com protegir les seves dades i el seu codi de canvis dinàmics que un atacant podria introduir.

No obstant això, cal assenyalar alguns defectes, com que aquestes tècniques varien constantment, que certs problemes en la implementació de les tècniques de protecció poden ser explotats, o que en determinats casos aquestes proteccions poden donar falsos positius.

Hardening-Wrapper

Es recomana per a més informació consultar les pàgines man de `hardening-wrapper` i `hardening-check`. En el cas de Debian, és recomanable: <https://wiki.debian.org/hardening>.

4. SELinux

La seguretat tradicional dins del sistema s'ha basat en les tècniques DAC (*discretionary access control*, control d'accés discrecional), en què normalment cada programa disposa de control complet sobre els accessos als recursos. Si un determinat programa (o l'usuari) decideix fer un accés incorrecte (per exemple, deixant dades confidencials en obert, per desconeixement o per mal funcionament), no hi ha res que li ho impedeixi. Així, en DAC, un usuari té control complet sobre els objectes que li pertanyen i els programes que executa. El programa executat disposarà dels mateixos permisos que l'usuari que l'està executant. Així, la seguretat del sistema dependrà de les aplicacions que s'estiguin executant i de les vulnerabilitats que poguessin tenir, o del programari maliciós que poguessin incloure i afectarà, en especial, els objectes (altres programes, fitxers o recursos) a què l'usuari tingui accés. En el cas de l'usuari *root*, això comprometria la seguretat global del sistema.

D'altra banda, les tècniques MAC (*mandatory access control*, control d'accés obligatori) desenvolupen polítiques de seguretat (definides per l'administrador) en les quals el sistema controla completament els drets d'accés que es concedeixen a cada recurs. Per exemple, amb permisos (de tipus UNIX) podem donar accés a fitxers, però mitjançant polítiques MAC tenim control addicional per a determinar a quins fitxers es permet accedir explícitament a un procés, i quin grau d'accés volem concedir. Es fixen contextos, en els quals s'indiquen en quines situacions un objecte pot accedir a un altre objecte.

SELinux [Nsab] és un component recent de tipus MAC inclòs en la branca 2.6.x del nucli que les distribucions van incloure progressivament: Fedora / Red Hat el duen habilitat per defecte (tot i que és possible canviar-lo durant la instal·lació) i en Debian és un component opcional. SELinux aplica polítiques de seguretat de tipus MAC i permet disposar d'un accés de permisos més fi que els tradicionals permisos d'arxiu UNIX. Per exemple, l'administrador podria permetre que s'afegissin dades a un arxiu de registre, però no reescriure'l o truncar-lo (tècniques utilitzades habitualment per atacants per a esborrar les pistes dels accessos). En un altre exemple, es podria permetre que programes de xarxa s'enllacessin a un port (o ports) que els calgués i, tot i així, denegar l'accés a altres ports (aquesta tècnica podria permetre controlar i limitar l'accés d'alguns troians o portes secretes). SELinux va ser desenvolupat per l'agència NSA dels Estats Units, amb aportacions de diverses companyies per a sistemes UNIX i lliures, com Linux i BSD. Es va alliberar l'any 2000 i des de llavors s'ha anat integrant en diferents distribucions GNU/Linux.

Amb SELinux disposem d'un model de domini-tipus, en què cada procés corre en un context de seguretat i qualsevol recurs (fitxer, directori, sòcol, etc.) té un

Enllaços d'interès

En els enllaços següents teniu alguns recursos sobre SELinux:

- <http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/selinux-guide>,
- <http://www.nsa.gov/research/selinux/index.shtml>,
- <http://fedoraproject.org/wiki/SELinux>,
- <http://selinux.sourceforge.net/>.

tipus associat amb ell. Hi ha un conjunt de regles que indiquen quines accions poden efectuar-se en cada context sobre cada tipus. Un avantatge d'aquest model context-tipus és que les polítiques que es defineixin es poden analitzar (hi ha eines) per a determinar quins fluxos d'informació estan permesos; per exemple, per a detectar possibles vies d'atac o si la política és prou completa per a cobrir tots els accessos possibles.

Es disposa del que es coneix com la *base de dades de polítiques de SELinux* (*SELinux policy database*), que controla tots els aspectes de SELinux: determina quins contextos pot utilitzar cada programa per a executar-se i especifica a quins tipus de cada context pot accedir.

En SELinux cada procés del sistema té un context compost de tres parts: una identitat, un rol i un domini. La identitat és el nom del compte de l'usuari, o bé `system_u` per a processos de sistema o `user_u` si l'usuari no disposa de polítiques definides. El rol determina quins contextos estan associats. Per exemple, `user_r` no té permès tenir el context `sysadm_t` (domini principal per a l'administrador de sistema). Així, un `user_r` amb identitat `user_u` no pot obtenir de cap manera un context `sysadm_t`. Un context de seguretat s'especifica sempre amb aquests tres valors, com per exemple:

```
root:sysadm_r:sysadm_t
```

que és el context per a l'administrador del sistema, en defineix la identitat, el rol i el context de seguretat.

Per exemple, en una màquina amb SELinux activat (una Fedora en aquest cas) podem veure amb l'opció `-Z` del `ps` els contextos associats als processos :

```
# ps -ax -Z
```

LABEL	PID	TTY	STAT	TIME	COMMAND
system_u:system_r:init_t	1	?	Ss	0:00	init
system_u:system_r:kernel_t	2	?	S	0:00	[migration/0]
system_u:system_r:kernel_t	3	?	S	0:00	[ksoftirqd/0]
system_u:system_r:kernel_t	4	?	S	0:00	[watchdog/0]
system_u:system_r:kernel_t	5	?	S	0:00	[migration/1]
system_u:system_r:kernel_t	6	?	SN	0:00	[ksoftirqd/1]
system_u:system_r:kernel_t	7	?	S	0:00	[watchdog/1]
system_u:system_r:syslogd_t	2564	?	Ss	0:00	syslogd -m 0
system_u:system_r:klogd_t	2567	?	Ss	0:00	klogd -x
system_u:system_r:irqbalance_t	2579	?	Ss	0:00	irqbalance
system_u:system_r:portmap_t	2608	?	Ss	0:00	portmap
system_u:system_r:rpcd_t	2629	?	Ss	0:00	rpc.statd
user_u:system_r:unconfined_t	4812	?	Ss	0:00	/usr/libexec/gconfd-2 5
user_u:system_r:unconfined_t	4858	?	Sl	0:00	gnome-terminal

```

user_u:system_r:unconfined_t 4861 ? S 0:00 gnome-pty-helper
user_u:system_r:unconfined_t 4862 pts/0 Ss 0:00 bash
user_u:system_r:unconfined_t 4920 pts/0 S 0:01 gedit
system_u:system_r:rpcd_t 4984 ? Ss 0:00 rpc.idmapd
system_u:system_r:gpm_t 5029 ? Ss 0:00 gpm -m /dev/input/mice -t exps2
user_u:system_r:unconfined_t 5184 pts/0 R+ 0:00 ps ax -Z
user_u:system_r:unconfined_t 5185 pts/0 D+ 0:00 Bash

```

I amb `ls` amb l'opció `-Z` podem veure els contextos associats a fitxers i directors:

```

# ls -Z
drwxr-xr-x josep josep user_u:object_r:user_home_t Desktop
drwxrwxr-x josep josep user_u:object_r:user_home_t proves
-rw-r--r-- josep josep user_u:object_r:user_home_t yum.conf

```

I des de la consola podem conèixer el nostre context actual amb:

```

$ id -Z
user_u:system_r:unconfined_t

```

Pel que fa al mode de funcionament, SELinux en presenta dos, anomenats *permissive* i *enforcing*. En *permissive* es permeten els accessos no autoritzats, però són auditats en els registres corresponents (normalment, directament sobre `/var/log/messages` o bé, depenent de la distribució, amb l'ús de `audit` en `/var/log/audit/audit.log`). En *enforcing* no es permet cap tipus d'accés que no permetin les polítiques definides. També pot desactivar-se SELinux per mitjà del fitxer de configuració (habitualment a `/etc/selinux/config`), col·locant `SELINUX=disabled`.

Cal anar amb compte amb l'activació i desactivació de SELinux, especialment amb l'etiquetatge de contextos en els fitxers, ja que en períodes en què s'activi o desactivi es poden perdre etiquetes (perquè el sistema no estarà actiu) o simplement poden no ser etiquetats. Així mateix, la realització de còpies de seguretat del sistema de fitxers ha de tenir en compte que cal conservar les etiquetes de SELinux.

Un altre possible problema que cal tenir en compte és el gran nombre de regles de política de seguretat que poden arribar a existir i que poden provocar limitacions en el control dels serveis. Davant d'un tipus determinat de mal funcionament, en primer lloc cal determinar que no sigui precisament SELinux allò que està impeding el funcionament, per una limitació massa estricta de seguretat (vegeu el subapartat 4.2. de crítica a SELinux) o a causa d'opcions que no esperàvem tenir activades (poden requerir canviar la configuració dels booleans, com veurem).

Pel que fa a la política usada, SELinux suporta dos tipus diferents: *targeted* i *strict*. En *targeted* la majoria de processos operen sense restriccions, i només serveis específics (alguns dimonis –*daemons*) es posen en diferents contextos de seguretat que són confinats a la política de seguretat. En *strict* tots els processos són assignats a contextos de seguretat i confinats a polítiques definides, de manera que qualsevol acció és controlada per les polítiques definides. En principi aquests són els dos tipus de polítiques definits generalment, però l'especificació resta oberta a incloure'n més.

Un cas especial de política és la MLS (*multilevel security*, seguretat multinivell), que és una política multinivell de tipus *strict*. La idea és definir, dins de la mateixa política, diferents nivells de seguretat, i els contextos de seguretat tenen associats un camp addicional de nivell d'accés. Aquest tipus de política de seguretat (com MLS) sol ser utilitzat en organitzacions governamentals i militars, on hi ha organitzacions jeràrquiques amb diferents nivells d'informació privilegiada, nivells d'accés general i capacitats d'acció diferents en cada nivell. Per a obtenir algunes certificacions de seguretat (concedides o avalades per certes organitzacions) per als sistemes operatius, cal disposar de polítiques de seguretat d'aquest tipus.

Es pot definir quin tipus de política s'usarà a `/etc/selinux/config`, variable `SELINUXTYPE`. La política corresponent, i la seva configuració, normalment estarà instal·lada en els directoris `/etc/selinux/SELINUXTYPE/`. Per exemple, sol trobar-se en el subdirectori `policy` el fitxer binari de la política compilada (que és el que es carrega en el nucli en inicialitzar SELinux).

4.1. Arquitectura

L'arquitectura de SELinux està formada pels components següents:

- Codi en l'àmbit del nucli
- La biblioteca compartida de SELinux
- La política de seguretat (la base de dades)
- Eines

Examinem algunes consideracions sobre cada component:

- El codi de nucli monitora l'activitat del sistema, i assegura que les operacions sol·licitades estiguin autoritzades sota la configuració de polítiques de seguretat del SELinux actual; així, no permet les operacions no autoritzades i normalment genera entrades en el registre de les operacions denegades. El codi actualment es troba integrat en els nuclis 2.6.x, mentre que en els anteriors s'ofereix com a sèrie de pedaços.
- Gairebé la majoria d'utilitats i components SELinux no directament relacionats amb el nucli fan ús de la biblioteca compartida (que s'anomena `libselinux1.so`), que facilita una API per a interaccionar amb SELinux.

- La política de seguretat és la que està integrada en la base de dades de regles de SELinux. Quan el sistema arrenca (amb SELinux activat), carrega el fitxer binari de política, que habitualment resideix en la ruta següent: `/etc/security/selinux` (tot i que pot variar segons la distribució).

El fitxer binari de polítiques es crea a partir d'una compilació (via `make`) dels fitxers font de polítiques i alguns fitxers de configuració.

Algunes distribucions (com per exemple Fedora) no instal·len les fonts per defecte, que solen trobar-se a `/etc/security/selinux/src/policy` o a `/etc/selinux`. Normalment, aquestes fonts consisteixen en diversos grups d'informació:

- Els fitxers relacionats amb la compilació, `makefile` i *scripts* associats.
- Els fitxers de la configuració inicial, usuaris i rols associats.
- Els fitxers de `Type-enforcement`, que contenen la majoria de les sentències del llenguatge de polítiques associat a un context particular. Cal tenir en compte que aquests fitxers són enormes, típicament de desenes de milers de línies, amb la qual cosa pot aparèixer el problema de trobar fallades o definir canvis en les polítiques.
- Els fitxers que serveixen per a etiquetar els contextos dels fitxers i directoris durant la càrrega o en determinats moments.
- Les eines: inclouen ordres usades per a administrar i utilitzar SELinux, versions modificades d'ordres estàndard Linux i eines per a l'anàlisi de polítiques i el desenvolupament.

Vegem, d'aquest últim punt (les eines típiques de què solem disposar), algunes de les ordres principals:

Nom	Utilització
<code>chcon</code>	Etiqueta un fitxer específic o un conjunt de fitxers amb un context específic.
<code>checkpolicy</code>	Fa diverses accions relacionades amb les polítiques, incloent-hi la compilació de les polítiques a binari; típicament es crida des de les operacions de <code>makefile</code> .
<code>getenforce</code>	Genera missatges amb el mode actual de SELinux (<i>permissive</i> o <i>enforcing</i>). O bé desactivat, si és el cas.
<code>getsebool</code>	Obté la llista de booleans, és a dir, la llista d'opcions <i>on/off</i> per a cada context associat a un servei o opció general del sistema.
<code>newrole</code>	Permet a un usuari la transició d'un rol a un altre.
<code>runn_init</code>	S'utilitza per a activar un servei (<i>start</i> , <i>stop</i>), garantint que es faci en el mateix context que quan s'arrenca automàticament (amb <i>init</i>).
<code>setenforce</code>	Canvia de mode a SELinux: 0 <i>permissive</i> , 1 <i>enforcing</i> .
<code>setfiles</code>	Etiqueta directoris i subdirectoris amb els contextos adequats; s'utilitza habitualment en la configuració inicial de SELinux.
<code>setstatus</code>	Obté l'estat del sistema amb SELinux.

D'altra banda, es modifiquen algunes ordres GNU/Linux amb funcionalitats o opcions noves, com ara `cp`, `mv`, `install`, `ls`, `ps`, etc., per exemple modificant en els fitxers l'etiqueta per a associar el context de seguretat (com vam poder comprovar amb l'opció `-Z` de `ls`). `Id` es modifica per a incloure l'opció de mostrar el context actual de l'usuari. I en `ps` vam veure que incloïa una opció per a visualitzar els contextos de seguretat actuals dels processos.

A més a més, es modifiquen alguns altres programes comuns per a donar suport a SELinux com:

- `cron`: modificat per a incloure els contextos per a les tasques en execució per `cron`.
- `login`: modificat perquè col·loqui el context de seguretat inicial per a l'usuari quan entra en el sistema.
- `logrotate`: modificat per a preservar el context dels *logs* quan estiguin recopilats.
- `pam`: modificat per a col·locar el context inicial de l'usuari i per a usar l'API SELinux i obtenir accés privilegiat a la informació de contrasenyes.
- `ssh`: modificat per a col·locar el context inicial de l'usuari quan entra en el sistema.
- Diversos programes addicionals que modifiquen `/etc/passwd` o també `/etc/shadow`.

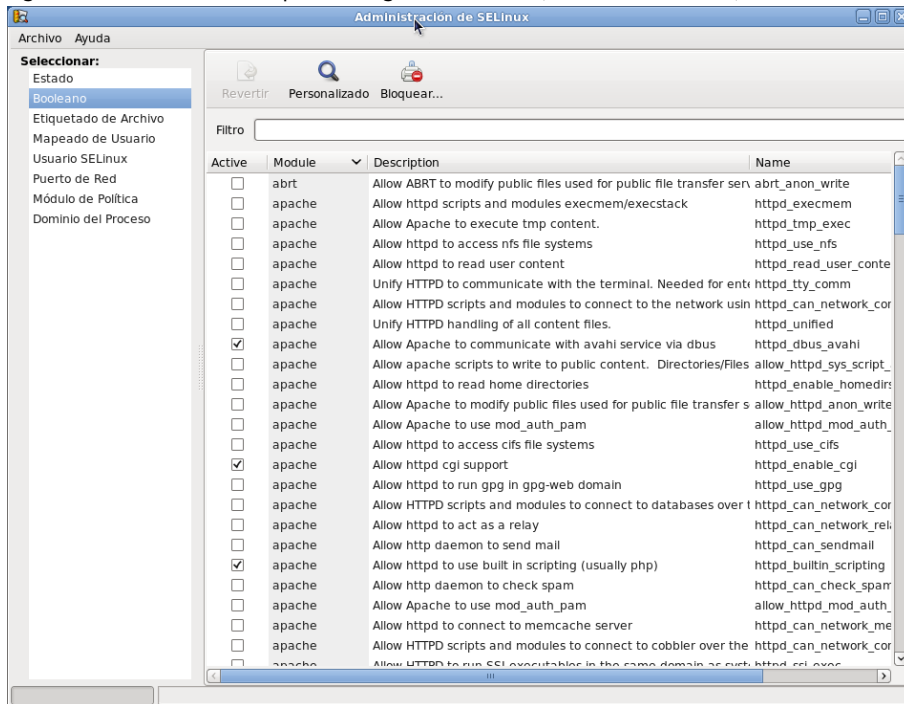
En algunes distribucions també s'inclouen eines per a la gestió de SELinux, com les `setools(-gui)`, que duen diverses eines de gestió i anàlisi de les polítiques. Així, com a eina específica per a controlar els contextos associats als diferents serveis suportats per SELinux en la distribució, en Fedora l'eina `system-config-securitylevel` disposa d'un apartat per a la configuració de SELinux, com podem observar en la figura 1.

En la figura s'observa la configuració de booleans per a diversos serveis i opcions genèriques, entre les quals el servidor web. També podem obtenir aquesta llista amb `getsebool -a`, i amb les ordres `setsebool/togglesebool` podem activar/desactivar les opcions.

En Fedora, per exemple, trobem suport de booleans, entre d'altres, per a `cron`, `ftp`, `httpd` (Apache), `dns`, `grub`, `lilo`, `nfs`, `nis`, `cups`, `pam`, `ppd`, `samba`, proteccions contra accessos incorrectes a la memòria dels processos, etc.

La configuració de booleans permet la personalització de la política SELinux en temps d'execució. Els booleans són utilitzats en SELinux com a valors condicionals de les regles de la política usada, fet que permet modificacions de la política sense necessitat de carregar una nova política.

Figura 1. Interfície en Fedora per a configuració SELinux (examinant booleans)



4.2. Crítica

Alguns administradors i experts en seguretat han criticat SELinux específicament, perquè és massa complex de configurar i administrar. S'argumenta que, per la complexitat intrínseca que té, fins i tot usuaris experimentats poden cometre errors, cosa que deixaria la configuració de SELinux no segura o inservible i el sistema, vulnerable. Això és discutible fins a cert punt, ja que tot i que tinguéssim SELinux mal configurat, encara seguirien actius els permisos UNIX i altres eines, i SELinux no permetria una operació que els permisos originals ja no permetessin. De fet, podem veure'l com una etapa addicional de seguretat més estricta.

També hi ha factors de rendiment que es poden veure afectats, a causa de l'enorme grandària de les polítiques; poden disminuir les prestacions com a conseqüència del gran ús de memòria i del temps inicial de càrrega i, en algun cas, del processament de les regles. Cal pensar que estem parlant pràcticament d'un sistema de més de 10.000 regles en la política. I encara pot ser un nombre més gran si escollim una política de tipus *strict*, amb la necessitat d'especificar absolutament totes les opcions que s'han de controlar. Normalment, el processament per política en format binari i l'ús de booleans per a inhabilitar regles permet un ús més eficient del sistema.

Un altre aspecte que sol molestar els administradors és el problema addicional de determinar, davant d'un mal funcionament específic, quin n'és l'origen o causa inicial. És habitual que finalment ens trobem que el problema provenia d'una configuració excessivament restrictiva (potser per desconeixement de l'administrador) de SELinux per a un determinat servei. En darrera instàn-

cia, cal assenyalar l'ampli suport per a la seguretat que proporciona SELinux i que, com a administradors, hem de ser conscients de les capacitats i perills de qualsevol nova tècnica que utilitzem.

4.3. Algunes alternatives

Davant la complexitat de SELinux i algunes deficiències del model (complexitat de polítiques o mala configuració), s'han produït algunes alternatives (o en alguns casos complements) a la utilització de SELinux. Hem de destacar en particular Grsecurity, una sèrie de pedaços per al nucli Linux que intenten millorar algunes deficiències. També cal destacar AppArmor, que algunes distribucions estan incloent com a alternativa a SELinux.

Grsecurity, per contra, s'ofereix com una sèrie de pedaços de nucli publicats posteriorment a l'establiment d'una versió estable del nucli, i es poden obtenir de la seva pàgina web per a certes versions de nucli, a més de paquets ja preparats per a algunes distribucions.

Grsecurity inclou diferents prestacions, com un model d'accés a recursos mitjançant gestió de rols (tant locals com remots), gestionats mitjançant una eina denominada *gradm*; auditació de les operacions internes al nucli, i prevenció de codi arbitrari en aquest (de fet fa ús d'un pedaça de nucli denominat PaX, que ofereix algunes d'aquestes proteccions). Això evita possibles errors de tipus desbordament de *buffer*, *heap* o *stack* (tècniques aprofitades per *exploits* del kernel). També inclou notificacions d'alertes de seguretat basades en IP de l'usuari. Els pedaços oferts per al nucli s'apliquen com a pedaços al codi font del nucli i permeten activar, durant la configuració del nucli prèvia a la seva compilació, diferents opcions de protecció del codi del nucli relacionades amb les prestacions esmentades. A més, una vegada el nucli està configurat per al seu suport, l'eina *gradm* permet gestionar els rols i les tasques o proteccions associades a cada rol i als usuaris que incloguem en cadascun.

Grsecurity és un bon complement o substitut d'alguns punts febles de SELinux, ja que ofereix una interfície més simple d'usar i inclou algunes prestacions de seguretat addicionals.

Una altra alternativa, que està sorgint en algunes distribucions com Ubuntu i SuSe, és AppArmor, que permet a l'administrador associar a cada programa un perfil de seguretat que restringeixi les capacitats del programa. Es pot establir un perfil individual de permisos o bé es pot fer servir un mode d'autoaprenentatge basat en violacions de perfil registrades i habilitades pels usuaris o no.

Bàsicament, es compon d'un mòdul de nucli per a la gestió de perfils d'aplicacions, basat, igual que SELinux, en LSM (*Linux Security Modulis*), juntament amb diferents utilitats. Mitjançant aquestes utilitats podem establir permisos

Grsecurity

<http://grsecurity.net>

Enllaç d'interès

Sobre l'administració de Grsecurity pot consultar-se <http://en.wikibooks.org/wiki/Grsecurity>.

AppArmor en Ubuntu

Una alternativa a SELinux utilitzada en Ubuntu és AppArmor: <https://help.ubuntu.com/14.04/serverguide/apparmor.html>.

dels programes per a l'accés al sistema de fitxers i als dispositius. La major diferència amb SELinux és que les polítiques de seguretat no s'apliquen amb etiquetes als fitxers, sinó respecte a polítiques aplicables als *pathnames*. AppArmor s'integra en el nucli com a mòdul LSM disponible a partir de les versions 2.6.36 del nucli.

De fet, en les branques actuals del kernel es troba el *framework* denominat LSM (*Linux Security Modules*), que suporta una varietat de mòduls de seguretat per a evitar el favoritisme envers algun dels esmentats. Ara mateix, en LSM, per a les implementacions de seguretat de tipus MAC (*mandatory access control*), estan acceptats com a part del kernel, des de les branques 2.6, els models d'AppArmor, SELinux, Smack i TOMOYO (a més de Yama, que podria incorporar-s'hi en el futur).

Sobre mòduls LSM en el kernel

Podeu llegir comentaris dels diferents models d'LSM en [Overview of LSM](#).

5. Seguretat en xarxa

5.1. Client de serveis

Com a clients de serveis, bàsicament ens hem d'assegurar que no posem en perill els nostres usuaris (o que es posin en perill ells tots sols) pel fet d'usar serveis insegurs. Cal evitar l'ús de serveis que no utilitzin xifratge de dades i contrasenyes (FTP, Telnet, correu no segur) i utilitzar en aquest cas tècniques de connexió xifrada, com SSH i SSL/TSL. Així mateix, hem d'assegurar-nos que trobem en el sistema els programes client, i les biblioteques que aquests fan servir (com les esmentades prèviament) en les versions més actualitzades. En alguns moments crítics, davant de possibles vulnerabilitats de tipus *Oday*, pot ser fins i tot necessari actualitzar serveis, clients o biblioteques, independentment de la distribució, per a assegurar-nos que el nostre sistema no és vulnerable, mentre no es disposi de solució a les vulnerabilitats detectades.

Un altre punt important que s'ha de tenir en compte es refereix a la possible substitució de servidors per altres de falsos, o bé a tècniques de segrest de sessions. En aquests casos hem de disposar de mecanismes d'autenticació robustos que ens permetin verificar l'autenticitat dels servidors (per exemple, SSH i SSL/TSL tenen alguns d'aquests mecanismes). També hauríem de verificar la xarxa a la recerca d'intrusos, perills potencials, intents d'intrusió fallits o intents de substitució de servidors, a més de fer servir polítiques correctes de filtratge de paquets mitjançant tallafocs (*firewalls*) que ens permetin donar sortida als nostres paquets de dades vàlides i usar els servidors adequats, alhora que controlem els paquets d'entrada que rebem com a resposta.

5.2. Servidor: inetd i xinetd

La configuració dels serveis de xarxa [Mou02], tal com hem vist, es pot fer des de diversos llocs [Ray01, Hat08, Pen]:

- A `/etc/inetd.conf` o el directori equivalent `/etc/xinetd.d`: aquests fitxers de configuració de serveis estan associats al que es coneix com a *superservidors*, ja que controlen diversos serveis fills i les seves condicions d'arrencada. El servei `inetd` és utilitzat en Debian (tot i que cada vegada menys), i `xinetd`, en Fedora (en Debian, `xinetd` es pot instal·lar opcionalment en substitució d'`inetd`).
- Servidors iniciats en arrencada: segons el nivell d'execució (*runlevel*) tindrem una sèrie de serveis arrencats. L'arrencada s'originarà en el directori

Clients i serveis insegurs

Com a clients de serveis, haurem d'evitar l'ús de serveis insegurs.

Cas HeartBleed

Un cas de vulnerabilitat crítica que va afectar les biblioteques `openssl` durant el 2014:
<http://heartbleed.com/>

Nota Systemd

En noves distribucions, el control de serveis s'està migrant, majoritàriament, des d'`inetd` i `xinetd` al concepte de *targets* basats en `Systemd`. Observeu:
<http://0pointer.de/blog/projects/security.html>

associat al nivell d'execució; per exemple, en Debian el nivell per defecte és el 2, els serveis arrencats ho seran des del directori `/etc/rc2.d`, segurament amb enllaços als *scripts* continguts a `/etc/init.d`, en els quals s'executarà amb els paràmetres `start`, `stop`, `restart`, segons correspongui.

- Altres serveis de tipus RPC: s'usen, per exemple, en NIS i NFS associats a crides remotes entre màquines. Amb l'ordre `rpcinfo -p` es pot examinar quins hi ha.

Altres fitxers de suport (amb informació útil) són els següents: `/etc/services`, que està format per una llista de serveis locals o de xarxa coneguts, juntament amb el nom del protocol (`tcp`, `udp` o altres) que es fa servir en el servei i el port que utilitza; `/etc/protocols`, que és una llista de protocols coneguts, i `/etc/rpc`, que és una llista de possibles servidors RPC, juntament amb els ports (`rpc`) usats. Aquests fitxers vénen amb la distribució i són una mena de base de dades que utilitzen les ordres i les eines de xarxa per a determinar els noms de serveis, protocols o `rpc` i els seus ports associats. Cal destacar que són fitxers més o menys històrics que no han de contenir obligatòriament totes les definicions de protocols i serveis. També es poden buscar diferents llistes de ports coneguts a Internet.

Una de les primeres accions que haurà de fer l'administrador serà deshabilitar tots els serveis que no estigui utilitzant o no tingui previst utilitzar; en aquest sentit caldrà documentar-se sobre l'ús dels serveis [Mou02] i quin programari pot necessitar-los [Neu].

En el cas de `/etc/inetd.conf`, n'hi ha prou de comentar la línia del servei que cal deshabilitar, col·locant-hi un “#” com a primer caràcter de la línia.

En l'altre model de serveis, utilitzat per defecte en Fedora (i opcional en Debian), el `xinetd`, la configuració es troba en el fitxer `/etc/xinetd.conf`, on es configuren alguns valors per defecte de control de registres i, després, la configuració de cada servei fill es fa mitjançant un fitxer dins del directori `/etc/xinetd.d`. En cada fitxer es defineix la informació del servei, equivalent a la que apareix a `inetd.conf`; en aquest cas, per a desactivar un servei solament cal posar una línia `disable = yes` dins del fitxer del servei. `xinetd` té una configuració més flexible que `inetd`, ja que separa la configuració dels diferents serveis en diferents arxius, i té força opcions per a limitar les connexions a un servei, en nombre o en capacitat. Tot plegat permet un control més eficient del servei i, si el vam configurar adequadament, podrem evitar alguns dels atacs per denegació de servei (DoS o DDoS).

Quant al tractament dels serveis dels nivells d'execució des d'ordres de la distribució, ja hem esmentat diverses eines en la unitat d'administració local que permeten habilitar o deshabilitar serveis. També hi ha eines gràfiques com `ksysv` de KDE o el `system-config-services` i `ntsysv` en Fedora (en Debian són recomanables `sysv-rc-conf`, `rcconf` o `bum`). I a una escala inferior, podem anar al nivell d'execució que vulguem (`/etc/rcx.d`) i desactivar els

serveis que ens interressi canviant les `S` o `K` inicials de l'*script* per un altre text: un mètode seria, per exemple, canviar `S20ssh` per `STOPS20ssh` i ja no arrencarà; la propera vegada, quan tornem a necessitar-lo, esborrarem el prefix i el tornarem a tenir actiu. Malgrat que aquest mètode és possible, és més recomanable utilitzar una sèrie d'eines simples que proporcionen les distribucions per a col·locar, treure o activar un servei determinat. Per exemple, ordres com `service` i `chkconfig` en Fedora, o similars en Debian, com `update-rc.d` i `invoke-rc.d`.

Un altre aspecte és el tancament de serveis no segurs. Tradicionalment, en el món UNIX s'havien utilitzat serveis de transferència d'arxius com FTP, de connexió remota com Telnet i ordres d'execució (d'inici de sessió o de còpia) remotes, moltes de les quals començaven amb la lletra *r* (per exemple, `rsh`, `rcp`, `rexec`, etc.). Altres perills potencials són els serveis `finger` i `rwhod`, que permetien obtenir informació des de la xarxa dels usuaris de les màquines; aquí el perill rau en la informació que podia obtenir un atacant i que li podia facilitar molt el treball. Tots aquests serveis no s'haurien de fer servir actualment, a causa dels perills que impliquen. Pel que fa al primer grup:

- FTP, Telnet: en les transmissions per xarxa no xifren les contrasenyes, de manera que qualsevol pot obtenir les contrasenyes de serveis o els comptes associats (per exemple, mitjançant un detector).
- `rsh`, `rexec`, `rcp`: tenen, a més, el problema que, en algunes condicions, ni tan sols necessiten contrasenyes (per exemple, si es fa des de llocs validats en el fitxer `.rhosts`), amb la qual cosa són novament insegurs i deixen grans portes obertes.

L'alternativa és fer servir clients i servidors segurs que suportin el xifratge dels missatges i l'autenticació dels participants. Hi ha alternatives segures en els servidors clàssics, però actualment la solució més freqüent és l'ús del paquet OpenSSH (que pot combinar-se també amb OpenSSL per a entorns web). OpenSSH ofereix solucions basades en les ordres `ssh`, `scp` i `sftp`, que permeten substituir els antics clients i servidors (s'utilitza un dimoni anomenat *sshd*). L'ordre `ssh` permet les antigues funcionalitats de Telnet, `rlogin` i `rsh`, entre d'altres, i `scp` seria l'equivalent segur de `rcp`, i `sftp`, de l'FTP.

Pel que fa a SSH, també cal tenir la precaució d'usar `ssh` en la versió 2. La primera versió té algunes vulnerabilitats conegudes; cal anar amb compte a instal·lar OpenSSH i, si no necessitem la primera versió, instal·lar només suport per a `ssh2` (en el fitxer de configuració `/etc/ssh/sshd_config` vegeu l'opció `Protocol`).

A més a més, la majoria de serveis que deixem actius en les nostres màquines s'haurien de filtrar posteriorment per mitjà de tallafocs, per a garantir que no fossin utilitzats o atacats per persones a les quals no anaven destinats.

6. Eines de seguretat: detecció de vulnerabilitats i intrusions

Amb els sistemes de detecció d'intrusos [Hat08] (IDS) es vol fer un pas més. Una vegada haguem pogut configurar més o menys correctament la nostra seguretat, el pas següent consistirà en una detecció i prevenció activa de les intrusions.

Els sistemes IDS creen procediments d'escolta i generació d'alertes en detectar situacions sospitoses, és a dir, busquen símptomes de possibles accidents de seguretat.

Hi ha sistemes basats en la informació local que, per exemple, recopilen informació dels registres del sistema, vigilen canvis en els fitxers de sistema o bé en les configuracions dels serveis típics. Altres sistemes estan basats en xarxa i intenten verificar que no es produeixen comportaments estranys com, per exemple, casos de suplantació, en què hi ha falsificacions d'adreces conegudes; controlen el trànsit sospitosos i possibles atacs de denegació de servei, detectant trànsit excessiu cap a determinats serveis i controlant que no hi ha interfícies de xarxa en mode promiscu (síntoma de *sniffers* o capturadores de paquets).

Alguns exemples d'eines IDS serien:

- Logcheck: verificació de *logs*.
- TripWire: estat del sistema mitjançant sumes md5 aplicades als arxius.
- Portsentry: protegeixen contra escanejos de ports i detecten indicis d'activitat sospitosa.
- AIDE: una versió lliure de TripWire.
- Snort: IDS de verificació d'estat d'una xarxa completa.

Algunes de les eines de seguretat es poden considerar també eines d'atac. Per tant, es recomana provar aquestes eines sobre màquines de la nostra xarxa local o privada. Mai no s'han de portar a terme atacs de prova amb adreces IP a tercers, ja que aquests podrien prendre-ho com a intrusions i demanar-nos responsabilitats, demanar-les al nostre ISP o avisar les autoritats competents perquè ens investiguin o tanquin el nostre accés.

Detecció d'intrusos

Els sistemes IDS ens permeten la detecció a temps d'intrusos fent servir els nostres recursos o explorant els nostres sistemes en cerca de fallades de seguretat.

A continuació, comentem breument algunes de les eines i alguns dels usos que se'ls pot donar:

1) Logcheck: una de les activitats d'un administrador de xarxa és verificar diàriament (més d'una vegada per dia) els arxius *log* per a detectar possibles atacs/intrusions o esdeveniments que puguin donar indicis sobre aquestes qüestions. Aquesta eina selecciona (dels arxius *log*) informació condensada de problemes i riscos potencials i després envia aquesta informació al responsable, per exemple, a través d'un correu. El paquet inclou utilitats per a executar-se de manera autònoma i recordar l'última entrada verificada per a les subsegüents execucions. La llista d'arxius que cal monitorar s'emmagatzema a `/etc/logcheck/logcheck.logfiles` i la configuració per defecte és adequada (si no es va modificar gran part de l'arxiu `/etc/syslog.conf`). Logcheck pot funcionar en tres modalitats: *paranoid*, *server* i *workstation*. La primera és molt detallada i hauria de limitar-se a casos específics com, per exemple, *firewalls*. La segona (per defecte) és la recomanada per a la majoria de servidors, i l'última és l'adequada per a estacions d'escriptori. Aquesta eina permet una configuració total dels filtres i de les sortides, si bé pot ser complicat reescriure'ls. Les regles es poden classificar en les següents: intent d'intrusió (*cracking*), emmagatzemades a `/etc/logcheck/cracking.d/`; les d'alerta de seguretat, emmagatzemades a `/etc/logcheck/violations.d/`; i les que són aplicades a la resta de missatges. [Her13]

2) Tripwire: aquesta eina manté una base de dades de sumes de comprovació dels fitxers importants del sistema. Pot servir com a sistema IDS preventiu. Ens serveix per a portar a terme una foto del sistema, poder comparar després qualsevol modificació introduïda i comprovar que no hagi estat malmès per un atacant. L'objectiu aquí és protegir els arxius de la pròpia màquina, evitar que es produeixin canvis com, per exemple, els que podria haver provocat un *rootkit*. Així, quan tornem a executar l'eina, podem comprovar els canvis respecte a l'execució anterior. Cal triar un subconjunt d'arxius que siguin importants en el sistema o font de possibles atacs. Hi ha altres eines lliures de funcionament equivalent, entre aquestes una possibilitat és AIDE. Tripwire (<http://sourceforge.net/projects/tripwire/>) és una eina que ajudarà l'administrador notificant possibles modificacions i canvis en arxius per a evitar possibles danys (majors). Aquesta eina compara les diferències entre els arxius actuals i una base de dades generada prèviament per a detectar canvis (insercions i esborrat), la qual cosa és molt útil per a detectar possibles modificacions d'arxius vitals, com per exemple, d'arxius de configuració.

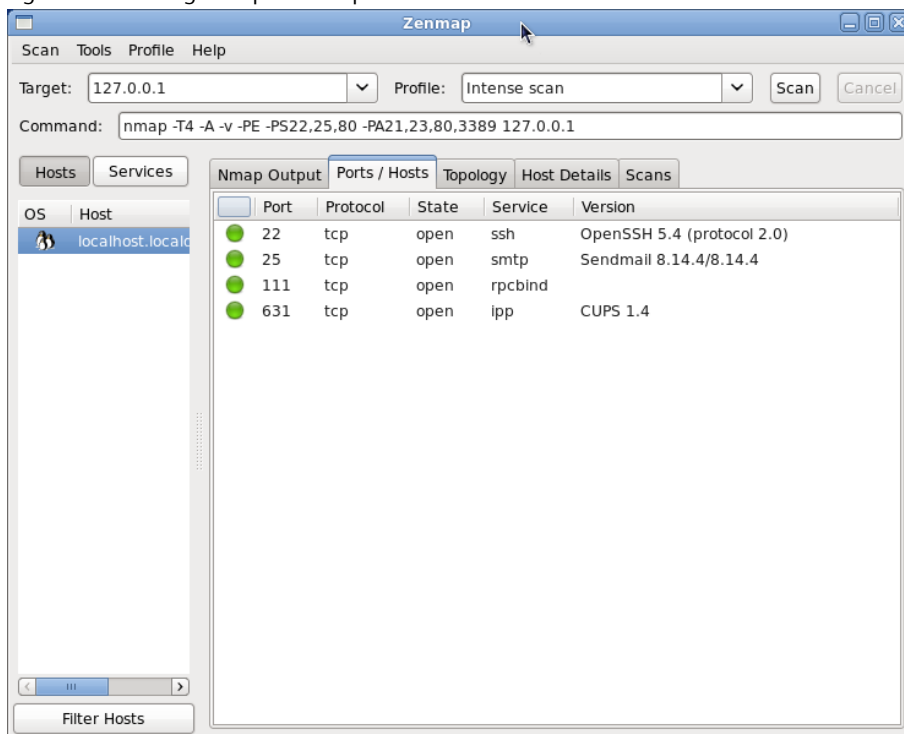
3) Portsentry: forma part d'un conjunt d'eines* que proporcionen serveis de seguretat de nivell de *host* per a GNU/Linux. PortSentry, LogSentry i Hostsentry protegeixen contra escanejos de ports i detecten indicis d'activitat sospitosa.

*<http://sourceforge.net/projects/sentrytools/>

4) Nmap [Insb]: és una eina d'escaneig de ports per a xarxes (figura 2). Pot escanejar des de màquines individuals a segments de xarxa. Permet diversos tipus d'escaneig de ports, depenent de les proteccions que tingui el sistema.

També té tècniques que permeten determinar el sistema operatiu que fan servir les màquines remotes. Pot emprar diferents paquets de TCP i UDP per a provar les connexions. Hi ha algunes interfícies gràfiques per a KDE i Gnome, i alguna amb bastants possibilitats, com Zenmap.

Figura 2. Interfície gràfica per a Nmap durant una anàlisi de serveis locals



5) tcpdump: es tracta d'una eina molt potent que està en totes les distribucions i que ens servirà per a analitzar els paquets de xarxa. Aquest programa permet l'abocament del trànsit d'una xarxa i pot analitzar la majoria de protocols utilitzats avui dia (IPv4, ICMPv4, IPv6, ICMPv6, UDP, TCP, SNMP, AFS, BGP, RIP, PIM, DVMRP, IGMP, SMB, OSPF, NFS, entre d'altres).

Donada la importància d'analitzar cap a on van els paquets i d'on vénen, mostrarem algunes ordres habituals de tcpdump:

```
# tcpdump
paràmetres per defecte -v o -vv per al nivell d'informació mostrada, -q sortida ràpida
# tcpdump -D
interfícies disponibles per a la captura
# tcpdump -n
mostra IP en lloc d'adreces
# tcpdump -i eth0
captura el trànsit d'eth0
# tcpdump udp
només els paquets UDP
# tcpdump port http
només els paquets del port 80 (web)
# tcpdump -c 20
solament els 20 primers paquets
```

```
# tcpdump -w capture.log
envia les dades a un arxiu
# tcpdump -r capture.log
llegeix les dades d'un arxiu
```

Les dades capturades no són text, per la qual cosa es pot utilitzar aquesta mateixa eina per a la seva lectura o bé fer-ne servir una altra, com Wireshark, per a portar a terme la seva lectura i descodificació. Amb `tcpdump host www.site.com` només examinem els paquets que continguin aquesta adreça.

```
tcpdump src 192.168.1.100 and dst 192.168.1.2 and port ftp
```

Per a mostrar els paquets FTP que vagin de 192.168.1.100 a 192.168.1.2.

```
tcpdump -A          mostra el contingut dels paquets.
```

6) Wireshark (abans denominat *Ethereal*): és un analitzador de protocols i captura el trànsit de la xarxa (actua com *sniffer*). Permet visualitzar el trànsit capturat, veure estadístiques i dades dels paquets individuals i agrupar-los, ja sigui per origen, destinació, ports o protocol. Pot fins i tot reconstruir el trànsit d'una sessió sencera d'un protocol TCP.

7) Snort [Sno]: és un dels millors sistemes IDS en GNU/Linux, que permet fer anàlisis de trànsit en temps real i guardar registres dels missatges. Permet portar a terme anàlisis dels protocols i cerques per patrons (protocol, origen, destinació, etc.). Pot usar-se per a detectar diversos tipus d'atacs. Bàsicament, analitza el trànsit de la xarxa per a detectar patrons que puguin correspondre a un atac. El sistema utilitza una sèrie de regles per a anotar la situació (*log*), produir un avís (*alert*) o descartar la informació (*drop*).

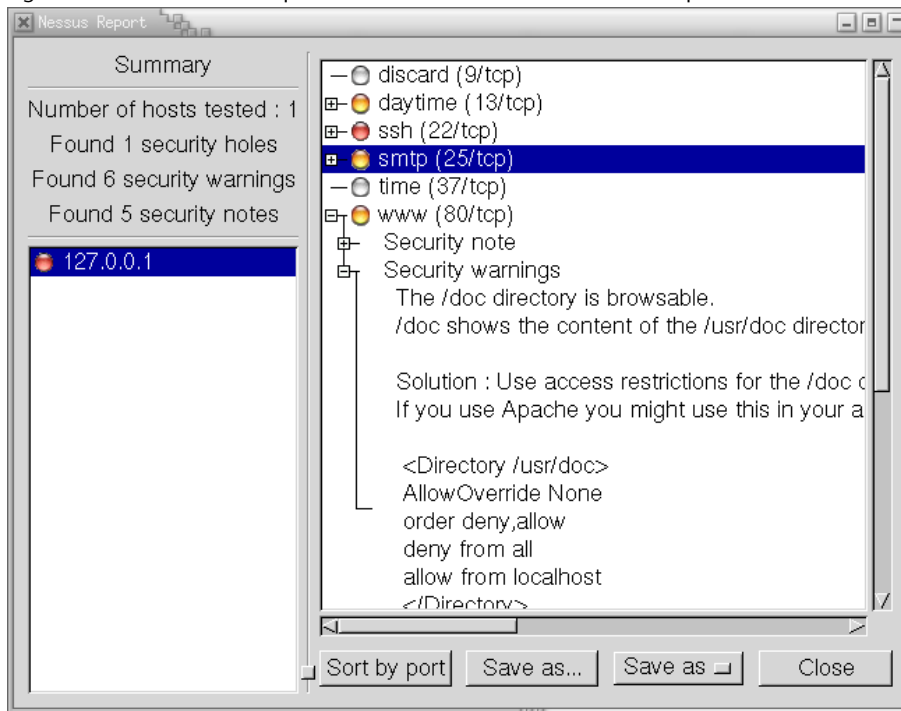
Instal·lació Snort

Snort és un sistema IDS altament recomanable. Per a la instal·lació detallada, podeu consultar les *Setup Guides* disponibles en <https://www.snort.org/documents>.

8) Nessus [Nes]: és un detector de vulnerabilitats conegudes, que prova diferents tècniques d'intrusió i assessora sobre com millorar la seguretat per a les detectades. És un programa modular que inclou més d'11.000 connectors (*plugins*) per a les diferents anàlisis. Utilitza una arquitectura client/servidor, amb un client gràfic que mostra els resultats i el servidor que porta a terme les diferents comprovacions sobre les màquines. Té capacitat per a examinar xarxes senceres i genera els resultats en forma d'informes que poden exportar-se a diferents formats (per exemple, HTML). Fins a l'any 2005, Nessus 2 era una eina lliure, però la companyia va decidir convertir-se en propietària en la versió Nessus 3. Encara en GNU/Linux, es pot trobar i seguir utilitzant Nessus 2 (figura 3), que es manté amb llicència GPL i una sèrie de connectors que es van actualitzant. Nessus 3+ (i versions posteriors) com a eina propietària per GNU/Linux és més potent, àmpliament utilitzada i una de les eines més populars de seguretat. Normalment es manté lliure de cost una versió amb els connectors menys actualitzats que la versió de pagament. I encara es manté Nessus 2 en llicència GPL, actualitzant-la sovint. També ha sorgit

una *fork* lliure a partir de Nessus 2, denominada OpenVAS, que està disponible pràcticament per a la majoria de distribucions GNU/Linux. Més endavant, analitzarem alguna de les seves possibilitats.

Figura 3. Client de Nessus 2 que mostra l'informe de vulnerabilitats i les possibles solucions.



Podem trobar moltes més eines de seguretat disponibles. Un bon lloc per a examinar és <http://sectools.org>, on els creadors del Nmap van mantenir una llista d'eines populars votades pels usuaris al llarg de diversos anys. Ara aquesta llista és una mica antiga, però es mantenen actualitzades les referències a les eines, les quals, a dia d'avui, són perfectament útils i de les més utilitzades.

Passarem ara, en les seccions següents, a analitzar diferents eines amb més detall i alguns casos d'ús. Algunes d'aquestes són bastant útils en aspectes generals de detecció de vulnerabilitats i intrusos.

6.1. OpenVAS

L'eina OpenVAS (*Open Vulnerability Assessment System*) és un conjunt d'eines que treballen a l'uníson per a córrer tests sobre màquines clients usant una base de dades d'*exploits* coneguts. La idea és saber en quina situació es troben els clients provats contra una sèrie d'atacs coneguts.

En aquest cas, utilitzarem una distribució Debian com a sistema base per a provar els clients corresponents. Comencem instal·lant el repositori necessari per a una Debian 7.x (wheezy):

(Nota: vigileu els salts de línia indicats en aquestes línies d'ordres amb \; haurien de ser una única línia sense aquest caràcter, ni espais afegits.)

```
# echo "deb http://download.opensuse.org/repositories/security:/OpenVAS\
:/UNSTABLE:/v6/Debian_7.0/ ." >> /etc/apt/sources.list
# wget http://download.opensuse.org/repositories/security:/OpenVAS\
:/UNSTABLE:/v6/Debian_7.0/Release.key
# apt-key add ./Release.key
# apt-get update
```

Així instal·lem el repositori d'OpenVAS per Debian, afegint-lo com a font de paquets, important el seu *hash* (del repositori), i afegint-lo al sistema perquè puguem autenticar els paquets del repositori. Seguidament, procedim a un *update* del sistema de paquets APT. I procedirem a instal·lar els paquets necessaris per als diferents components d'OpenVAS, paquets necessaris per a la generació d'informes i alguns de necessaris per a la creació de certificats posteriors:

```
# apt-get -y install greenbone-security-assistant openvas-cli openvas-manager \
openvas-scanner openvas-administrator sqlite3 xsltproc rsync
# apt-get -y install texlive-latex-base texlive-latex-extra \
texlive-latex-recommended htmldoc
# apt-get -y install alien rpm nsis fakeroot
```

El següent bloc d'ordres, totes com a usuari *root*, s'encarrega de generar els certificats necessaris per al lloc OpenVAS, preparar les bases de dades que s'utilitzaran (sincronitzant-la per Internet amb els servidors d'OpenVAS i els seus *plugins* disponibles), parar els *daemons* d'OpenVAS, reiniciar el *daemon* principal d'OpenVAS (es fa la sincronització de dades necessàries) i afegir usuari com a *admin*, per la qual cosa ens preguntaran el *password* que desitgem. Posteriorment, s'aturarà el *daemon* general d'OpenVAS i es reiniciaran els serveis. Aquest bloc següent pot executar-se directament com a *root* sobre una Debian 7.x (amb els passos anteriors prèviament portats a terme). Cal tenir en compte que la sincronització de dades amb els servidors OpenVAS pot portar un temps considerable:

```
test -e /var/lib/openvas/CA/cacert.pem || openvas-mkcert -q
openvas-nvt-sync
test -e /var/lib/openvas/users/om || openvas-mkcert-client -n om -i
/etc/init.d/openvas-manager stop
/etc/init.d/openvas-scanner stop
openvassd
openvasmd --rebuild
openvas-scapdata-sync
openvas-certdata-sync
test -e /var/lib/openvas/users/admin || openvasad -c add_user -n admin -r Admin
killall openvassd
sleep 15
/etc/init.d/openvas-scanner start
/etc/init.d/openvas-manager start
/etc/init.d/openvas-administrator restart
/etc/init.d/greenbone-security-assistant restart
```

Una vegada efectuats tots aquests passos, podem accedir a l'eina en el nostre sistema arrencant el navegador web i obrint l'URL:

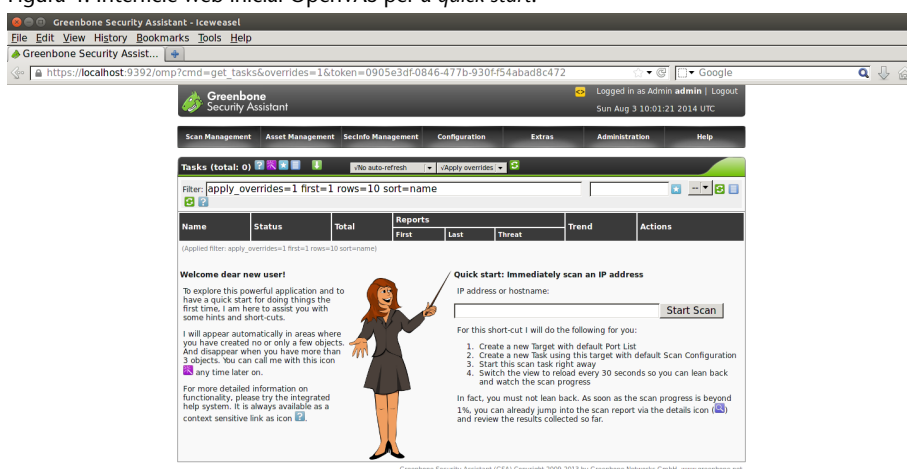
<https://localhost:9392/>

Quan obrim l'adreça esmentada, tindrem un procés de *login*, que superarem mitjançant l'usuari *login* creat.

En accedir, tindrem la interfície web disponible, on mitjançant un *wizard* podrem directament indicar la IP o DNS del sistema on farem l'escàner de vulnerabilitats.




Com sempre, cal assenyalar de nou que utilitzarem l'eina per a escanejar un *host* propi i no contra tercers, que podrien prendre's l'escaneig com un intent d'atac.

Figura 4. Interfície web inicial OpenVAS per a *quick start*.



Una vegada que l'escàner hagi progressat, podem accedir al resultat mitjançant les icones de l'apartat *Actions* de la interfície associada.

Figura 5. Vulnerabilitats detectades amb el seu nivell de risc.

Report	Threat	Scan Results					Actions
		High	Medium	Low	Log	Details Page	
Sun Aug 3 10:05:56 2014 Done	High	6	10	1	74	0	  

Greenbone Security Assistant (GSA) Copyright 2009-2013 by Greenbone Networks GmbH, www.greenbone.net

Al final, disposarem de l'informe en forma d'una llista agregada de potencials vulnerabilitats detectades, amb un nivell de *thread* i les vulnerabilitats associades a cada nivell de perill. Podem de nou examinar els detalls de l'informe (icona de lupa) per a disposar de la possibilitat de visualitzar-lo o descarregar-lo (en diversos formats).

En la interfície de l'informe (*Report Summary*), podem filtrar els resultats per diversos criteris, i a sota podem examinar cada incidència segons el seu nivell de perill, verificant l'explicació de la vulnerabilitat i indicacions per a la seva possible solució.

Figura 6. Una vulnerabilitat detectada.

Port summary	
Service (Port)	Threat
http-alt (8080/tcp)	High
https (443/tcp)	High
xmltec-xmlmail (9091/tcp)	High
general/tcp	Medium
http (80/tcp)	Medium
smtp (25/tcp)	Medium

Security Issues reported	
High (CVSS: 6.8) NVI: Apache Tomcat servlet/JSP container default files (OID: 1.3.6.1.4.1.25623.1.0.12085)	http-alt (8080/tcp)
<p>Default files, such as documentation, default Servlets and JSPs were found on the Apache Tomcat servlet/JSP container.</p> <p>Remove default files, example JSPs and Servlets from the Tomcat Servlet/JSP container.</p> <p>These files should be removed as they may help an attacker to guess the exact version of Apache Tomcat which is running on this host and may provide other useful information.</p> <p>The following default files were found :</p> <pre>/examples/servlets/index.html /examples/jsp/snp/snoop.jsp /examples/jsp/index.html</pre>	

Així doncs, ja disposem d'una instal·lació bàsica d'OpenVAS funcional per a controlar vulnerabilitats dels nostres servidors, i podem usar-la com a mitjà per a perfeccionar problemes de seguretat del nostre entorn.

Hi ha diverses opcions extra disponibles per a perfeccionar la instal·lació, entre les quals es troben tasques de programes d'escaneig, generar informes automàticament i enviar alertes per correu. Podeu explorar les opcions d'interfície per a obtenir una configuració més detallada i així optimitzar l'ús d'OpenVAS.

6.2. Denyhosts

En aquest cas, tractem amb un altre problema habitual. Es tracta de l'administració dels accessos a través d'SSH i de l'administració de comptes remots dels usuaris. El servei d'SSH és propens a rebre atacs de diverses tipologies, com ara:

- 1) Atacs controlats cap a ports, típicament el TCP 22, on s'allotja per defecte.
- 2) Atacs contra usuaris Linux predeterminats o de sistema, buscant comptes de tipus convidat que estiguessin amb *passwords* per defecte, o algunes contrasenyes d'usuaris associats a serveis (per exemple, bases de dades, CMS, control de versions o altres) que hagin estat instal·lades amb els seus comptes per defecte, o contrasenyes provinents de defecte.
- 3) Atacs contra usuari *root*, mitjançant diccionaris, guiats o simplement per la força bruta.

El servei SSH en si mateix ofereix certes possibilitats (mitjançant el fitxer de configuració del seu *daemon* `/etc/ssh/sshd_config`) per a mitigar aquests atacs, com canviar el port per defecte del servei, o en el plànol dels usuaris,

intentar mitigar els problemes mitjançant una selecció correcta de les seves contrasenyes i promocionant polítiques de canvi i ús de contrasenyes no trivials. A l'usuari *root* se li pot denegar, a més del seu accés en SSH (opció *PermitRootLogin no*), l'accés remot, la qual cosa no impedirà que l'atacant intenti aconseguir accés local mitjançant algun altre usuari. També podrem complementar serveis com SSH amb altres mesures: a més d'aquelles que veurem en l'apartat següent 7, amb *TCP wrappers*, amb l'ús de tallafocs (*firewalls*), o també inhabilitant l'ús de contrasenyes per a SSH i forçant així l'ús de claus pública/privada per als usuaris d'SSH.

Davant d'aquests problemes, Denyhosts ofereix la possibilitat de monitorar els *logs* d'SSH i impedir (*ban*) l'accés a usuaris que detecti que possiblement estan portant a terme un atac remot. Pot configurar-se per a permetre una sèrie d'intents de connexió i els temps entre aquests. Perquè Denyhosts funcioni correctament, el *daemon* d'SSHD ha d'estar compilat amb l'opció de *TCP wrappers* (comentarem aquesta tècnica en el següent apartat 7), encara que en el cas dels *daemons* instal·lats de paquets de la distribució ja ve així per defecte.

En el cas d'una distribució Debian (de manera semblant a una Fedora), el podem instal·lar amb:

```
# apt-get install denyhosts
```

I podem configurar els paràmetres en el fitxer */etc/denyhosts.conf*, un exemple de valors de certs paràmetres (hi ha moltes més opcions, en particular les relacionades amb correu de notificació mitjançant opcions *SMTP_*):

```
SECURE_LOG = /var/log/auth.log
BLOCK_SERVICE = sshd
DENY_THRESHOLD_INVALID = 5
DENY_THRESHOLD_VALID = 10
DENY_THRESHOLD_ROOT = 1
DENY_THRESHOLD_RESTRICTED = 1
HOSTNAME_LOOKUP=YES
```

Primer establim el lloc per a examinar el *log* corresponent als intents d'SSH (en aquest cas, en Debian és el fitxer esmentat), que serà el fitxer que monitorarà Denyhosts. Controlarem el bloqueig del servei *sshd* (*daemon ssh*) si es donen les circumstàncies (poden controlar-se altres serveis). En aquest exemple, permetem fins a 5 intents fallits de comptes no vàlids (no existents en el sistema), 10 intents fallits de comptes registrats en el sistema (en */etc/passwd*), i 1 intent sobre *root* (això té sentit, per exemple, si sabem que no està permès l'accés des de *root*); aquests seran els intents abans de bloquejar el *host* que els ha produït. En l'últim, es permet que es faci una cerca DNS per a l'IP que ha portat a terme la fallada per intents excessius.

Cal anar amb compte amb els valors de *Threshold* per a assegurar que complim uns mínims de possibilitats, ja que intents vàlids des de *hosts*, per oblits o fallades repetides, poden provocar que un *host* vàlid es bloquegi.

El control anterior dels *hosts* que es permeten i no es fa amb els fitxers `/etc/hosts.deny` i `/etc/hosts.allow`, per bloquejats i permesos; per exemple, si ens volem assegurar que el nostre *host* habitual no es bloquegi (sempre que disposem d'IP estàtica), podríem col·locar (en `/etc/hosts.allow`):

```
ALL: la nostra_ip
```

Posteriorment, podem reiniciar el servei amb:

```
# /etc/init.d/denyhosts restart
```

Els *hosts* que es bloquegin al llarg del temps acabaran incloent-se en l'arxiu `/etc/hosts.deny`. També poden purgar-se al cap d'un temps (hi ha opcions denominades *PURGE_* en la configuració de *denyhosts* que ho controlen de manera automàtica); podran purgar-se explícitament mitjançant:

```
# /etc/init.d/denyhosts stop
# denyhosts --purge
# /etc/init.d/denyhosts start
```

6.3. Fail2ban

Fail2Ban, disponible en Debian i Fedora, és similar en concepte a l'eina prèvia *Denyhosts*, però en lloc d'enfocar-se a SSH, en aquest cas, pot ser configurada per a monitorar qualsevol servei que escrigui intents de *login* en un fitxer de *log* i, en lloc de bloquejar, posar l'*host* culpable d'atacs en `/etc/hosts.deny` (encara que també pot configurar-se perquè actuï així). Fa els bloquejos per mitjà d'*iptables* (sistema de tallafocs per defecte del kernel Linux, com veurem en l'apartat 7).

Instal·lem amb:

```
# apt-get install fail2ban
```

I disposem posteriorment de la seva configuració per al nostre sistema en el directori `/etc/fail2ban`, en el qual trobarem el comportament per defecte en el fitxer `/etc/fail2ban/jail.conf`. En algunes versions recents, es recomana no editar aquest fitxer i disposar la configuració local en el directori

`/etc/fail2ban/jail.d` o en `/etc/fail2ban/jail.local`. Consulteu la pàgina man depenent de la versió disponible en la nostra distribució.

Algunes de les configuracions presents (com a exemples) es refereixen a:

- `ignoreip = 127.0.0.1/8` : si són IP de màquines conegudes que no volem que `fail2ban` bloquegi. En aquest cas, el propi *localhost* i adreces associades.
- `bantime = 600` : temps en segons en què una màquina és bloquejada si es detecta com a problemàtica.
- `findtime = 600` i `maxretry=3` : un *host* és bloquejat si abans que arribi a *findtime* segons ha generat *maxretry* intents fallits.

Després, per a cada servei trobem diversos camps (en la secció del fitxer de configuració `[servei]`); normalment, per defecte només es troba activat el servei d'SSH:

- `enabled`: activat el control del servei.
- `port`: port de servei que s'està utilitzant, normalment fent servir els noms de `/etc/services`.
- `filter`: filtres aplicats dels disponibles en `/etc/fail2ban/filter.d`. Aquests filtres serveixen per a enganxar els missatges d'esdeveniments dels *logs* examinats, per a detectar si corresponen o no al servei.
- `action`: la mesura que cal prendre per al bloqueig, per exemple *iptables-allports*, que bloquejarà els ports que faci servir el servei per al *host* problemàtic. Les *actions* que tenim disponibles les podem trobar en el directori `/etc/fail2ban/action.d/`.
- `logpath`: fitxer de *log* que es comprova per a detectar *logins* fallits.
- `maxretry`: si canviem en el servei el nombre màxim d'intents fallits.

Per exemple, en una Debian podríem copiar `/etc/fail2ban/jail.conf` en `/etc/fail2ban/jail.local` i ajustar els paràmetres locals generals com creguem convenient, i dins del fitxer cadascuna de les seccions corresponents al servei.

Posteriorment, reiniciem el servei:

```
# /etc/init.d/fail2ban restart
```

I si per exemple tenim *fail2ban* emetent *logs* a `/var/log/fail2ban.log`, podríem observar sortides com:

```
2014-08-24 18:49:09,466 fail2ban.actions: WARNING [apache] Ban 1.2.3.4
2014-08-24 19:08:33,213 fail2ban.actions: WARNING [ssh] Ban 1.2.3.4
```

I observar amb *iptables* si s'han produït bloquejos (en aquest cas, es visualitza allò produït en el servei SSH):

```
#iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source                destination            multiport dports ssh
fail2ban-ssh tcp  --  anywhere              anywhere
.....
Chain fail2ban-ssh (1 references)
target     prot opt source                destination
DROP      all  --  1.2.3.4              anywhere
RETURN    all  --  anywhere             anywhere
.....
```

si disposem dels *hosts* problemàtics bloquejats.

Algunes ordres útils per a examinar els *logs*, comprovar l'estatus general d'un servei particular o eliminar el bloqueig *iptables* d'una IP determinada:

```
# tail /var/log/fail2ban.log
# fail2ban-client status
# fail2ban-client status ssh
# iptables -D fail2ban-<CHAIN_NAME> -s <IP> -j DROP
```

En què en l'últim cas es desbloqueja una IP si el bloqueig es va produir per a *iptables*, i on normalment (tret que haguem canviat el tipus d'*action*), la *CHAIN_NAME* serà la corresponent al servei, en l'exemple anterior SSH (*fail2ban-ssh*).

7. Protecció mitjançant filtratge (*TCP wrappers* i tallafocs)

Els *TCP wrappers* [Mou02] són un programari que actua d'intermediari entre les peticions de l'usuari del servei i els dimonis dels servidors que ofereixen el servei. Moltes de les distribucions vénen ja amb els *wrappers* activats i podem configurar els nivells d'accés. Els *wrappers* se solen utilitzar en combinació amb *inetd* o *xinetd*, de manera que protegeixin els serveis que ofereixen.

El *wrapper* bàsicament substitueix el dimoni del servei per un altre que fa d'intermediari (anomenat *tcpd*, normalment a `/usr/sbin/tcpd`). Quan aquest rep una petició, en verifica l'usuari i l'origen, per determinar si la configuració del *wrapper* del servei permet utilitzar-lo o no. A més, incorpora facilitats per a generar registres o bé informar per correu dels possibles intents d'accés i després executa el dimoni adequat assignat al servei.

Per exemple, suposem l'entrada següent a *inetd*:

```
finger stream tcp nowait nobody /usr/etc/in.fingerd in.fingerd
```

Es canvia per:

```
finger stream tcp nowait nobody /usr/sbin/tcpd in.fingerd
```

De manera que, quan arribi una petició, serà tractada pel dimoni *tcpd*, que s'encarregarà de verificar-ne l'accés (per a més detalls, vegeu la pàgina *man tcpd*).

També existeix un mètode alternatiu de *TCP wrapper*, en què l'aplicació original es compila amb la biblioteca de *wrappers*. Llavors l'aplicació no cal que sigui a *inetd* i podrem controlar-la de la mateixa manera que en el primer cas, amb la configuració que comentem a continuació.

El sistema de *wrappers* es controla des del fitxer `/etc/hosts.deny`, on especifiquem quins serveis deneguem i a qui, per mitjà d'opcions, com un petit intèrpret d'ordres per a guardar la informació de l'intent, i des del fitxer `/etc/hosts.allow`, on solem col·locar el servei que utilitzarem, seguit de la llista de qui deixem entrar (més endavant, en el taller, en veurem un petit exemple). També existeixen les ordres *tcpdchk*, que comproven la configuració dels fitxers amfitrions (vegeu *man hosts_access* i *hosts_options*), per verificar que són correctes; és a dir, comproven la configuració. L'altra ordre útil és

Wrappers

Els *wrappers* permeten controlar la seguretat mitjançant llistes d'accés en l'àmbit dels serveis.

`tcpdmatch`, a la qual assignem un nom de servei i un possible client (usuari i/o amfitrió) i ens diu què faria el sistema davant d'aquesta situació.

7.1. Tallafocs

Un tallafoc (*firewall*) és un sistema o grup de sistemes que reforça les polítiques de control d'accés entre xarxes. El tallafoc pot estar implementat en programari, com una aplicació especialitzada que s'executa en un ordinador individual, o bé pot tractar-se d'un dispositiu especial dedicat a protegir un o més ordinadors.

En general, disposarem o bé de l'aplicació de tallafocs per a protegir una màquina concreta connectada directament a Internet (directament o per proveïdor), o bé podrem col·locar en la nostra xarxa una o diverses màquines dedicades a aquesta funció, de manera que protegeixin la nostra xarxa interna.

Tècnicament, la millor solució és disposar d'un ordinador amb dues o més targetes de xarxa que aïllin les diferents xarxes (o segments de xarxa) connectades, de manera que el programari de tallafocs de la màquina (o en el seu cas, un maquinari especial) s'encarregui de connectar els paquets de les xarxes i determinar quins poden passar i quins no, i a quina xarxa.

Aquest tipus de tallafocs sol combinar-se amb un encaminador (*router*) per a enllaçar els paquets de les diferents xarxes. Una altra configuració típica és la de tallafocs cap a Internet, per exemple amb dues targetes de xarxa: en una obtenim o proporcionem trànsit a Internet i en l'altra enviem o proporcionem el trànsit a la nostra xarxa interna, de manera que podem eliminar el trànsit que no va destinat a nosaltres, i també controlar el trànsit que es mou cap a Internet, per si no volem que es tingui accés a alguns protocols o bé sospitem que hi ha possibilitats de fuites d'informació a causa d'alguns atacs. Una tercera possibilitat és la màquina individual connectada amb una única targeta de xarxa cap a Internet, directament o bé a través d'un proveïdor. En aquest cas, només volem protegir la nostra màquina d'atacs d'intrusos, de trànsit no desitjat o de la possibilitat de robatori de dades.

És a dir, en aquests casos podem veure que el tallafoc, depenent de si és programari o no, de si la màquina té una o diverses targetes de xarxa o de si protegeix una màquina individual o una xarxa, pot tenir configuracions i usos diferents.

El tallafoc, en general, permet definir a l'usuari una sèrie de polítiques d'accés (quines són les màquines a les quals es pot connectar o quines poden rebre informació i el tipus d'informació que pot rebre) per mitjà del control dels ports TCP/UDP permesos d'entrada (*incoming*) o de sortida (*outcoming*). Algunes eines de gestió de tallafocs vénen amb una sèrie de polítiques preconfigurades, o només diuen si es vol un nivell de seguretat alt, mitjà o baix, o, finalment, permeten personalitzar les opcions totalment (màquines, protocols, ports, etc.).

Una altra tècnica de vegades relacionada és la NAT (*network address translation*, traducció d'adreces de xarxa). Aquesta tècnica proporciona una via per a ocultar les adreces IP usades en la xarxa privada i les oculta d'Internet, però hi manté l'accés des de les màquines. Un dels mètodes típics és el denominat *masquerading*. Si s'usa *NAT masquerading*, un o diversos dispositius en la xarxa poden aparèixer com una única adreça IP vistos des de fora. Això permet connectar diversos ordinadors a un únic dispositiu de connexió externa; per exemple, un cas d'encaminador ADSL a la llar permet connectar diverses màquines sense necessitat que el proveïdor ens proporcioni diferents adreces IP. Els encaminadors ADSL acostumen a oferir algun tipus de *NAT masquerading* i també possibilitats de tallafoc. Sol ser bastant comú utilitzar una combinació d'ambdues tècniques. En aquest cas, entra en joc, a més de la configuració de la màquina del tallafoc (els casos vistos abans), la configuració de la xarxa privada interna que volem protegir.

Seguretat a nivell de paquets

Els tallafocs permeten establir seguretat a nivell de paquets i en les connexions de comunicació.

7.2. Netfilter: iptables

El nucli Linux (a partir de versions 2.4.x) proporciona un subsistema de filtratge anomenat Netfilter [Net], que ofereix característiques de filtratge de paquets i també NAT. Aquest sistema permet fer servir diferents interfícies de filtratge, entre les quals la més utilitzada s'anomena *iptables*. L'ordre principal de control és *iptables*. Abans s'oferia un altre sistema de filtratge, anomenat *ipchains*, en els nuclis 2.2 [Gre] i el sistema tenia una sintaxi diferent (tot i que amb semblances conceptuals). En els nuclis 2.0 s'utilitzava un altre sistema anomenat *ipfwadm*. Aquí (i en els exemples posteriors) tractarem només amb Netfilter/*iptables* (és a dir, amb les possibilitats de tallafoc en nuclis de les versions 2.4/2.6/3.x). També comentarem noves propostes, com *nftables*, que serà el proper substitut de Netfilter/*iptables*, a partir d'algunes de les últimes branques kernel 3.x, i segurament el *firewall* per defecte en la branca 4.x i superiors. *nftables* té capes de compatibilitat amb *iptables*, amb la qual cosa segurament durant un temps veurem encara de manera majoritària regles de *firewall* compatibles amb *iptables*.

Respecte a Netfilter, com comentàvem, el seu principal component en aquests moments és *iptables*, que funciona com una eina de tallafoc (*firewall*), amb el suport en un nivell de kernel, fet que permet les accions de filtratge dels paquets i gestió de translació d'adreces de xarxa (NAT).

La interfície de l'ordre *iptables* permet fer les diverses tasques de configuració de les regles que afecten el sistema de filtratge: generació de registres, accions de preencaminament i postencaminament de paquets, NAT i reenviament (*forwarding*) de ports.

L'arrencada del servei es fa amb `/etc/init.d/iptables start`, si no estava ja configurada en el nivell d'execució.

Enllaç d'interès

Sobre Netfilter vegeu <http://www.netfilter.org>.

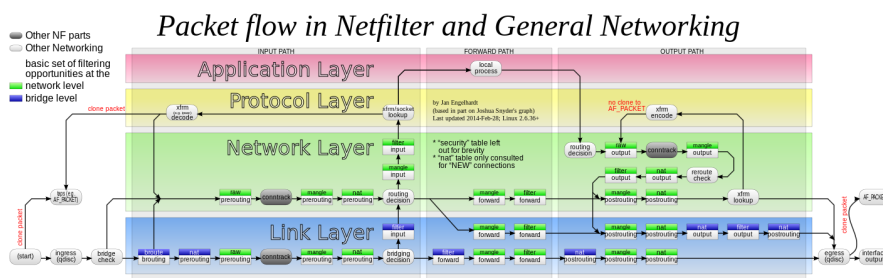
Elements d'iptables

iptables aporta diversos elements, com les taules, cadenes (*chains*) i les regles mateixes.

L'ordre `iptables -L` genera una llista de les regles actives en aquell moment en cada una de les cadenes. Si no s'han configurat prèviament, per defecte acostumen a ser l'acceptació de tots els paquets de les cadenes d'INPUT (entrada), OUTPUT (sortida) i FORWARD (reenviament).

El sistema d'iptables té com a nivell superior les taules. Cada una conté diferents cadenes que, al seu torn, contenen diferents regles. Les tres taules que hi ha són Filter, NAT i Mangle. La primera serveix per a les normes de filtratge, la segona, per a fer translació d'adreces dins d'un sistema que utilitzi NAT, i la tercera, menys usada, serveix per a especificar algunes opcions de control dels paquets i com es poden gestionar. Concretament, si estem amb un sistema directament connectat a Internet, utilitzarem, en general, només la taula Filter. Si el sistema és en una xarxa privada que ha de passar per un encaminador, passarel·la (*gateway*) o servidor intermediari (*proxy*), o una combinació d'aquests, segurament disposarem d'un sistema de NAT o IP *masquerading*; si estem configurant precisament la màquina que permet accés extern, haurem de tocar les taules NAT i la Filter. Si la màquina és en un sistema de xarxa privada, però és una de les màquines internes, serà suficient amb la taula Filter, tret que sigui un servidor el que faci NAT a un altre segment de xarxa.

Figura 7. Fluxos dels paquets de xarxa involucrats en la gestió de NetFilter/iptables



Més detall de la figura 7

La figura procedent de Wikipedia pot observar-se amb més detall en l'article: <http://en.wikipedia.org/wiki/iptables>

Si un paquet arriba al sistema, en el tallafoc implementat per iptables es mirarà primer si hi ha regles a la taula NAT, per si cal fer traduccions d'adreces cap a la xarxa interna (les adreces normalment no són visibles a l'exterior); després es miraran les regles de la taula Filter per a decidir si es deixaran passar els paquets o si no són per a nosaltres, i, si tenim regles `forward`, per a saber cap on els reenviem. En canvi, quan els nostres processos generen paquets, les regles `output` de la taula Filter controlen si els deixem sortir o no, i si hi hagués sistema NAT, les regles traduirien les adreces de manera que quedessin emmascarades. En la taula NAT sol haver-hi dues cadenes: `prerouting` i `postrouting`. En la primera, les regles han de decidir si cal fer algun encaminament del paquet i quina serà l'adreça de destinació. En la segona, es decideix finalment si el paquet es fa passar o no cap a l'interior (la xarxa privada, per exemple). També existeix una cadena `output` per al trànsit que es generi localment de sortida a la xarxa privada, ja que `prerouting` no ho controla (per a més detalls, es pot examinar *man iptables*).

Tot seguit comentarem alguns aspectes i exemples de configuració de la taula Filter*. La configuració típica de la taula Filter és d'una sèrie de regles que especifiquen què es fa dins d'una determinada cadena, com les tres anteriors (input, output o forward). Normalment, s'especifica:

*Per a les altres taules, es pot consultar la bibliografia associada.

```
iptables -A chain -j target
```

En què chain és input, output o forward, i target és la destinació que es donarà al paquet que es correspongui amb la regla. L'opció -A afegeix la regla a les existents.

Amb aquesta fase d'afegir regles cal prendre precaucions, ja que l'ordre importa. Cal col·locar les menys restrictives al principi, ja que, si primer posem una regla que elimini els paquets, malgrat que hi hagi una altra regla, no es tindrà en compte. L'opció -j permet decidir què farem amb els paquets, habitualment accept (acceptar-los), reject (rebutjar-los) o drop (simplement perdre'ls). És important la diferència entre reject i drop. Amb el primer, rebutgem el paquet i normalment informem l'emissari que hem rebutjat l'intent de connexió (normalment per un paquet de tipus ICMP). Amb el segon (drop), simplement perdem el paquet com si mai no hagués existit i no enviem cap tipus de resposta. Un altre target utilitzat és log, per a enviar el paquet al sistema de registres. Normalment, en aquest cas hi ha dues regles, una amb el log i una altra d'igual amb accept, drop o reject, per a permetre enviar al registre la informació de quins paquets han estat acceptats, rebutjats o perduts. Amb les opcions de generar-los en el tallafoc cal controlar-ne l'ús amb precisió, ja que són capaços, depenent de l'ambient de xarxa, de generar una enorme quantitat d'informació en els fitxers de registre.

Quan col·loquem la regla, també es pot utilitzar l'opció -I (insertar) per a indicar una posició, per exemple:

```
iptables -I INPUT 3 -s 10.0.0.0/8 -j ACCEPT
```

Que ens diu que es col·loqui la regla en la cadena input en tercera posició, i que s'acceptaran paquets (-j) que provenguin (amb font, o source, -s) de la subxarxa 10.0.0.0 amb màscara de xarxa 255.0.0.0. De manera semblant, amb -D podem esborrar un número de regla o la regla exacta, tal com s'especifica a continuació, esborrant la primera regla de la cadena o la regla que esmentem:

```
iptables -D INPUT 1
iptables -D INPUT -s 10.0.0.0/8 -j ACCEPT
```

També hi ha regles que permeten definir una política per defecte dels paquets (opció -P): es farà el mateix amb tots els paquets. Per exemple, se sol establir

a l'inici que es perdin tots els paquets per defecte, i s'habiliten després els que interessin. Moltes vegades també s'evita que es reenviïn paquets si no cal (si no actuem com a encaminador). Això podria posar-se així:

```
iptables -P INPUT DENY
iptables -P OUTPUT REJECT
iptables -P FORWARD REJECT
```

Tot plegat estableix unes polítiques per defecte que consisteixen a denegar l'entrada de paquets, no permetre sortir i no reenviar paquets. Ara es podran afegir les regles que fan referència als paquets que volem utilitzar, dient quins protocols, ports i orígens o destinacions volem permetre o evitar. Això pot ser difícil, ja que hem de conèixer tots els ports i protocols que utilitzen el nostre programari o els nostres serveis. Una altra tàctica seria deixar actius només aquells serveis que siguin imprescindibles i habilitar amb el tallafoc l'accés dels serveis a les màquines que ens interessin.

Alguns exemples d'aquestes regles de la taula Filter podrien ser:

```
iptables -A INPUT -s 10.0.0.0/8 -d 192.168.1.2 -j DROP
iptables -A INPUT -p tcp --dport 113 -j REJECT --reject-with tcp-reset
iptables -I INPUT -p tcp --dport 113 -s 10.0.0.0/8 -j ACCEPT
```

On:

- 1) Perdem els paquets que vinguin de 10.x.x.x amb destinació 192.168.1.2.
- 2) Rebutgem els paquets tcp amb destinació al port 113, i s'emet una resposta de tipus tcp-reset.
- 3) Acceptem els mateixos paquets que a 2), però que provenguin de 10.x.x.x.

Pel que fa als noms de protocols i ports, el sistema iptables fa servir la informació facilitada pels fitxers `/etc/services` i `/etc/protocols`, i es pot especificar la informació (de port i de protocol) de manera numèrica o per nom (cal anar amb compte, en aquest cas, que la informació dels fitxers sigui correcta i que no hagin estat modificats, per exemple, per un atacant).

La configuració d'iptables sol establir-se mitjançant crides consecutives a l'ordre `iptables` amb les regles. Això crea un estat de regles actives que poden consultar-se amb `iptables -L`. Si volem desar-les perquè siguin permanents, podem fer-ho, en Fedora, amb:

```
/etc/init.d/iptables save
```

I es desen a:

```
/etc/sysconfig/iptables
```

En Debian es pot fer, si existeix l'*script* anterior:

```
/etc/init.d/iptables save nom-regles
```

Si no s'ofereix suport directe per a desar o recuperar regles, sempre es pot amb les ordres `iptables-save` i `iptables-restore` rederigir-les a fitxers, per a desar o recuperar la configuració del tallafoc.

En el primer cas, cal estar ben segurs que existeixi prèviament el directori `/var/lib/iptables`, que és on es desen els fitxers; `nom-regles` serà un fitxer en el directori.

Amb `/etc/init.d/iptables load` podem carregar les regles (en Debian cal donar el nom del fitxer de regles o bé usar `iptables-restore`), tot i que Debian suporta uns noms per defecte de fitxers, que són `active` per a les regles normals (que s'utilitzaran en un `start` del servei), i `inactive` per a les que quedaran quan es desactivi el servei (quan es faci un `stop`). Una altra aproximació que es fa servir sovint és la de col·locar les regles en un fitxer *script* amb les crides `iptables` que calgui i cridar-les, per exemple, col·locant-les en el nivell d'execució necessari o amb un enllaç cap a l'*script* a `/etc/init.d`.

Per acabar, esmentarem un petit exemple d'allò que podria ser un fitxer complet d'`iptables` (el `#` indica comentaris):

```
*filter
# Permetre tot el trànsit de loopback (lo0) i denegar la resta de 127/8
-A INPUT -i lo -j ACCEPT
-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT
# Acceptar totes les connexions entrants prèviament establertes
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# Acceptar tot el trànsit sortint
-A OUTPUT -j ACCEPT
# Permetre HTTP i HTTPS des de qualsevol lloc
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT
# Permetre les connexions d'SSH
# Normalment utilitza el port 22, verificar-lo en el arxiu /etc/ssh/sshd_config.
-A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT
# Respondre al ping icmp
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
# Rebutjar tot el trànsit restant d'entrada
-A INPUT -j REJECT
-A FORWARD -j REJECT
COMMIT
```

En aquest exemple, només acceptem trànsit entrant de ping, http, https, ssh i bloquegem tot el trànsit restant d'entrada. Quant al trànsit de sortida, es deixa que surti tot sense restriccions. Es poden salvar les regles en un arxiu, carregar-les i, per exemple, des d'una altra màquina provar la connectivitat o executar

algun programa com nmap, que ens mostrarà els ports oberts en la màquina configurada amb iptables.

7.3. Paquets per a la gestió de tallafocs en les distribucions

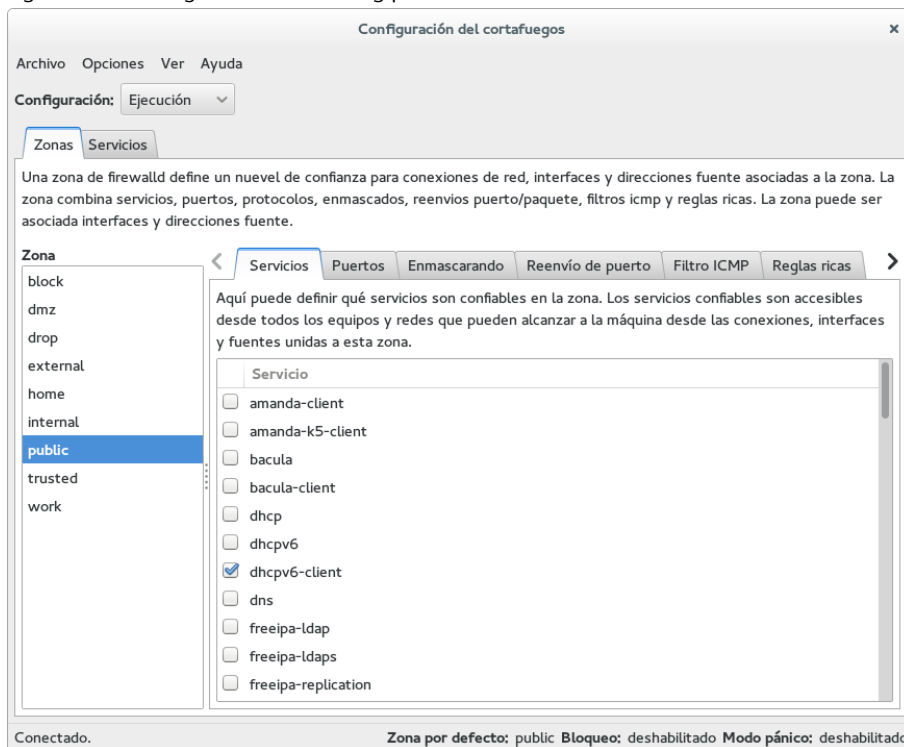
Quant a eines de configuració més o menys automatitzades per tallafocs, hi ha diverses possibilitats, però cal tenir en compte que no solen oferir les mateixes prestacions que la configuració manual d'iptables (que en la majoria de casos seria el procés recomanat). Algunes eines són:

- *system-configure-firewall* / *Lokkit*: en la distribució Fedora / Red Hat, fins fa poc era inicialment un gestor de *firewall* molt bàsic i només permetia triar a l'usuari el nivell de seguretat que desitjava (alt, mitjà o baix). Després, ensenyava els serveis que es veien afectats i podíem deixar passar o no el servei canviant la configuració per defecte. En les últimes versions, ha evolucionat bastant i ja inclou possibilitats avançades de ports que cal controlar per a cada servei. El mecanisme utilitzat per sota és iptables. És possible veure la configuració final de regles que porta a terme `/etc/sysconfig/iptables` que, al seu torn, és llegit pel servei iptables, que es carrega en arrencada o per parada o arrencada mitjançant `/etc/init.d/iptables`, amb les opcions `start` o `stop` (o les equivalents en Systemd). S'ofereix en versió gràfica o textual anomenada *system-config-firewall-tui*.
- En les versions recents de Fedora / Red Hat, s'ha inclòs un nou gestor per defecte de *firewall*, que encapsula el funcionament d'iptables i que substitueix el previ (*system-config-firewall*), denominat *FirewallD*. Aquest proporciona control dinàmic del *firewall* i permet definir múltiples zones de *firewall* per a gestionar les diverses xarxes a les quals estigui connectat el sistema. Permet mantenir, de manera simultània, configuracions permanents i en temps d'execució. L'antic control de *firewall* de Fedora era estàtic i requeria que, cada vegada que les regles canviaven, s'hagués de reiniciar el *firewall*. FirewallD permet aplicar les noves regles sense necessitat de reiniciar el *firewall* complet. Algunes ordres útils d'aquest cas que ens permeten obtenir l'estatus, els serveis suportats, habilitar un servei, obtenir les zones i les zones amb serveis habilitats són:

```
# firewall-cmd --state
# firewall-cmd --get-services
# firewall-cmd --add-service=http
# firewall-cmd --get-zones
# firewall-cmd --list-all-zones
```

Es disposa també de l'eina gràfica *firewall-config* per a portar a terme una configuració més còmoda. En entorn de terminal, es recomana veure el man corresponent a *firewall-cmd*, a causa del gran nombre d'opcions disponibles.

Figura 8. Interfície gràfica firewall-config per a FirewallID

**FirewallID**

Es recomana examinar.
<https://fedoraproject.org/wiki/firewalld>

- **fwbuilder**: una eina que permet construir les regles del tallafoc de manera gràfica. Es pot fer servir en diversos sistemes operatius (GNU/Linux tant Fedora com Debian, OpenBSD, MacOS), amb gestió de diferents tipus de tallafocs (incloent iptables).
- **firestarter**: una altra eina, ara obsoleta, però en distribucions antigues es feia servir com a eina gràfica (Gnome) per a la creació del tallafoc. Pràcticament maneja totes les possibilitats d'iptables, però així mateix disposa d'assistents que faciliten la construcció intuïtiva del tallafoc. Disposava, a més, d'un monitor en temps real per a detectar les intrusions.
- **ufw (Uncomplicated Firewall)**: com el seu nom indica, pretén ser un *firewall* d'ús simple, que pot trobar-se en les distribucions Debian i Ubuntu, basat en ús de línia d'ordres, que permet un ús bastant senzill (però suportant gran part de les possibilitats d'iptables). Una petita sessió d'exemple:

```
$ sudo ufw allow ssh/tcp
$ sudo ufw logging on
$ sudo ufw enable
$ sudo ufw status
Firewall loaded
```

To	Action	From
--	-----	----
22:tcp	ALLOW	Anywhere

També es disposa d'un paquet gràfic per al seu ús anomenat *Gufw*.

ufw

Lloc i eina gràfica:
<https://launchpad.net/ufw>
<http://gufw.org/>

Normalment, cada un d'aquests paquets utilitza un sistema de regles que desa en algun fitxer propi de configuració, i que sol arrencar com a servei o com a execució de *script* en el nivell d'execució per defecte. Cal tenir una cura especial amb la consistència de les regles iptables del tallafoc, ja que moltes d'aquestes utilitats no suporten sincronització bidireccional de les regles: o bé s'actualitza el tallafoc sempre mitjançant l'eina de línia d'ordres o gràfica, o bé es fa amb l'ordre `iptables`; si es fan canvis d'ambdues maneres, pot ser que aquests canvis no siguin consistents o es pot perdre la configuració del tallafoc. Es recomana fer còpies de seguretat periòdiques de la configuració del tallafoc (en especial si aquest té un cert grau de complexitat).

7.4. Netfilter: nftables

Nftables es presenta com un subprojecte de Netfilter per a la següent generació de tallafocs per a Linux, amb la intenció de substituir la infraestructura actual basada en iptables i proporcionar un nou *framework* per al filtratge de paquets (inclòs en l'espai de kernel, a partir de la versió 3.13), una nova utilitat *nft* d'usuari (en espai d'usuari) i una capa de compatibilitat amb iptables.

Algunes diferències destacables amb iptables:

- Sintaxi diferent, més clara.
- Les taules i cadenes són totalment configurables per l'usuari, a diferència d'iptables, que es basa en una estructura de taules predefinida.
- Major control de la correspondència de protocols. En iptables, algunes vegades eren necessàries extensions.
- Diverses accions en una sola regla. En iptables se'n podien haver de generar unes quantes.
- Els protocols de xarxa suportats poden actualitzar-se portant a terme modificacions en les aplicacions clients, sense necessitat d'actualitzar el kernel, evitant així problemàtiques amb versions de kernel congelades en certes distribucions.

Quant als camins que segueixen els paquets, continuen tenint la mateixa estructura (de *hooks*, en terminologia Netfilter): quan un paquet arriba al sistema, passa per un *prerouting*, i accedeix després a una decisió de *routing* (és un paquet d'entrada per al sistema o de reexpedició cap a uns altres?); en cas que fem *forwarding* en el nostre sistema (perquè funciona d'encaminador, per exemple amb `echo 1 > /proc/sys/net/ipv4/ip_forward`), el paquet arribarà al *postrouting* i sortirà del sistema, depenent de les opcions de *routing*. Si estem en el cas contrari, i el paquet era per al sistema, passarà a `INPUT`, serà processat pel sistema, que pot generar paquets de sortida, `OUTPUT`, i després

d'una decisió de *routing* (és per al propi sistema o per a enviament extern?) arribarà al *postrouting* per a sortir del sistema.

Per a l'ús d'*nftables*, el kernel ha de disposar del mòdul *nf_tables*, que pot estar disponible en el kernel (3.13+) o pot arribar-se a construir a partir dels fonts per kernels anteriors. Així, també es necessita el mòdul de la família, per exemple *nf_tables_ipv4*. Si els tenim disponibles, podem procedir a la seva càrrega amb *modprobe*.

Una vegada disposem dels mòduls idonis, podem procedir a fer algunes funcions bàsiques (com hem comentat, no hi ha taules o cadenes [*chains*] predefinides, per tant les haurem de crear):

```
Crear taula:
# nft add table ip filter

Llistar taules:
# nft list tables ip

Llistar cadenes associades a una taula (en principi no estaran creades):
# nft list table ip filter

Esborrar una taula:
# nft delete table filter

Flush d'una taula (alliberar regles de la taula):
# nft flush table ip filter

Afegir una cadena (chain) base
(relacionada amb un Netfilter Hook comentat abans),
aquesta cadena veu els paquets de la pila TCP/IP de Linux,
exemples d'input i output (si fos necessari)
i tercer cas una cadena no base.
Amb delete o flush en lloc d'add, esborrem el flush de les regles.
# nft add chain ip filter input { type filter hook input priority 0 \; }
# nft add chain ip filter output { type filter hook output priority 0 \; }
# nft add chain ip filter test

Regles: llistar-les
(opcions -n desactiva resolució noms, -nn resolució servei),
una regla per a un port concret,
una regla a una posició prèvia concreta dins de la cadena
guardar regles d'una cadena en un fitxer
recuperar-les a la cadena
# nft list table filter
# nft add rule filter output tcp dport ssh counter
# nft add rule filter output position 8 ip daddr 127.0.0.8 drop
# nft list table filter > filter-table
# nft -f filter-table
```

Finalment estan les accions disponibles, ja sigui acceptant paquets, perdent-los, rebutjant-los, fent NAT, etc. Vegem un breu exemple d'aquestes possibilitats. Per a la protecció simple d'una màquina d'escriptori, per a ipv4 i ipv6, podria ser el següent (aquest contingut és un fitxer amb la definició de les taules i regles bàsiques, que pot recuperar-se amb *nft -f fitxer* per a carregar el *firewall*.)

Accions nftables

Per a més detalls, es recomana consultar *Possible actions on packets* en:
http://wiki.nftables.org/wiki-nftables/index.php/main_page


```
# IPv4 filtering
table filter {
    chain input {
        table filter hook input priority 0;
        ct state established accept
        ct state related accept
        meta iif lo accept
        tcp dport ssh counter packets 0 bytes 0 accept
        counter packets 5 bytes 5 log drop
    }

    chain output {
        table filter hook output priority 0;
        ct state established accept
        ct state related accept
        meta oif lo accept
        ct state new counter packets 0 bytes 0 accept
    }
}

#IPv6 filtering
table ip6 filter {
    chain input {
        table filter hook input priority 0;
        ct state established accept
        ct state related accept
        meta iif lo accept
        tcp dport ssh counter packets 0 bytes 0 accept
        icmpv6 type { nd-neighbor-solicit, echo-request, \
nd-router-advert, nd-neighbor-advert } accept
        counter packets 5 bytes 5 log drop
    }

    chain output {
        table filter hook output priority 0;
        ct state established accept
        ct state related accept
        meta oif lo accept
        ct state new counter packets 0 bytes 0 accept
    }
}
}
```

7.5. Consideracions

Encara que disposem de tallafocs ben configurats, cal tenir present que no són una mesura de seguretat absoluta, ja que hi ha atacs complexos que poden saltar-se el control o falsejar dades que creïn confusió. A més, les necessitats de connectivitat modernes obliguen de vegades a crear programari que permeti la derivació o el pas (*bypass*) dels tallafocs:

- Tecnologies com IPP, protocol d'impressió utilitzat per CUPS, o el WebDAV, protocol d'autoria i actualització de llocs web, permeten passar per sobre (o cal que ho facin) de les configuracions dels tallafocs.
- Sovint s'utilitza (per exemple, els protocols anteriors i d'altres) una tècnica anomenada *tunelització* (*tunneling*), que bàsicament encapsula protocols no permesos sobre la base d'altres que sí que ho estan; per exemple, si un tallafoc permet només pas de trànsit HTTP (port 80 per defecte), és possible escriure un client i un servidor (cada un en un costat diferent del tallafoc) que parlin amb qualsevol protocol conegut per tots dos, però que a la xarxa

Nivells de seguretat

Mai no s'ha de confiar en un únic mecanisme o sistema de seguretat. Cal establir la seguretat del sistema amb diferents nivells.

es transformi en un protocol HTTP estàndard, de manera que el trànsit pugui creuar el tallafoc. El *tunneling* per ssh també s'utilitza per a establir diferents tipus de túnels, per a connexions, amb els protocols i ports d'ssh.

- Els codis mòbils per a web (ActiveX, Java i JavaScript) creuen els tallafocs (via web) i, per tant, és difícil protegir els sistemes si són vulnerables als atacs contra forats descoberts.

Així, tot i que els tallafocs són una solució molt bona a la majoria d'amenaques a la seguretat, sempre poden tenir vulnerabilitats i deixar passar trànsit que es consideri vàlid, però que inclogui altres fonts possibles d'atacs o vulnerabilitats.

En seguretat mai no s'ha de considerar (ni confiar en) una única solució i esperar que ens protegeixi de manera absoluta; cal examinar els diversos problemes, plantejar solucions que detectin els perills a temps i establir polítiques de prevenció que ens curin en salut, pel que pugui passar.

8. Anàlisi de registres

Si observem els fitxers de registre (*logs*) del sistema [Ray01, Fri02], podem fer-nos una idea ràpida de l'estat global del sistema i dels últims esdeveniments produïts, i detectar accessos (o intents d'accés) indeguts. Però també cal tenir present que els registres, si s'ha produït una intrusió real, poden haver estat netejats o falsejats. La majoria d'arxius de registre són al directori `/var/log`.

Molts dels serveis poden tenir registres propis, que normalment s'estableixen en la configuració (mitjançant el corresponent fitxer de configuració). La majoria solen utilitzar les facilitats de registre incorporades en el sistema Syslog, mitjançant el dimoni Syslogd. La seva configuració és en el fitxer `/etc/syslog.conf`. Aquesta configuració sol establir-se per nivells de missatges: existeixen diferents tipus de missatges segons la importància que tenen. Normalment hi ha els nivells `debug`, `info`, `err`, `notice`, `warning`, `crit`, `alert` i `emerg`, ordenats més o menys en funció de la importància dels missatges (de més baixa a més alta). Normalment, la majoria dels missatges es dirigeixen cap al registre `/var/log/messages`, però pot definir-se que cada tipus es distribueixi cap a fitxers diferents i també es pot identificar qui els ha originat: habitualment el nucli, el correu, missatges (*news*), el sistema d'autenticació, etc.

En conseqüència, cal examinar (i en tot cas adaptar) la configuració de Syslog per a saber en quins registres podem trobar o generar la informació. Un altre punt important és controlar-ne el creixement, ja que segons els que estiguin actius i les operacions (i serveis) que s'executin en el sistema, els registres poden créixer bastant. En Debian i Fedora es controla per mitjà de `logrotate`, un dimoni que s'encarrega periòdicament de fer còpies dels registres més antics i comprimir-les; se'n pot trobar la configuració general a `/etc/logrotate.conf`, però algunes aplicacions fan configuracions específiques que es poden trobar en el directori `/etc/logrotate.d`.

En els punts següents comentem alguns fitxers de registre que caldria tenir en compte (potser els més utilitzats):

1) `/var/log/messages`: és el fitxer de registre per defecte del dimoni Syslogd, però caldria revisar-ne la configuració, no fos cas que s'hagués canviat en una altra banda o que n'hi hagués més d'un. Aquest fitxer conté una gran varietat de missatges de diversos orígens (arrencada, diferents dimonis, serveis o el nucli mateix); tot allò que resulti anòmal s'hauria de verificar. En cas que s'hagi produït una intrusió, s'haurà de mirar al voltant de les dates de la intrusió en un determinat interval de temps, per a detectar possibles intents previs, la metodologia usada, primers avisos del sistema, etc.

Fitxers de registre

Cal tenir en compte que en la majoria d'atacs sofisticats reeixits es falsegen o s'esborren les dades dels fitxers de registre. Tot i així, depenent de la tècnica usada, es pot detectar informació útil sobre l'atac.

2) `/var/log/utmp`: aquest fitxer conté informació binària per a cada usuari que és actiu actualment. És interessant per a determinar qui és dins del sistema. L'ordre `who` utilitza aquest fitxer per a proporcionar aquesta informació.

3) `/var/log/wtmp`: cada vegada que un usuari entra en el sistema, en surt o es reinicia la màquina, es desa una entrada en aquest fitxer. És un fitxer binari del qual l'ordre `last` obté informació en què s'esmenta quins usuaris han entrat en el sistema o n'han sortit, quan s'ha originat la connexió i on. Pot ser útil per a buscar on (en quins comptes) s'ha originat una intrusió o per a detectar usos de comptes sospitosos. També hi ha una variant de l'ordre, anomenada `lastb`, que genera una llista dels intents d'inici de sessió que no s'han pogut validar correctament i es fa servir el fitxer `/var/log/btmp` (pot ser necessari crear-lo si no existeix). Aquests mateixos errors d'autenticació també solen enviar-se al registre `auth.log`. De manera semblant, l'ordre `lastlog` utilitza un altre fitxer `/var/log/lastlog` per a verificar quina va ser la darrera connexió de cada un dels usuaris.

4) `/var/log/secure`: sol utilitzar-se en Fedora per a enviar els missatges dels *tcp wrappers* (o dels tallafocs). Cada vegada que s'estableix una connexió a un servei d'`inetd`, o bé per a `xinetd` (amb la seva pròpia seguretat), s'afegeix un missatge de registre a aquest fitxer. Poden buscar-se intents d'intrusió en serveis que no s'utilitzen habitualment, o bé màquines no familiars que s'intenten connectar.

En el sistema de registres, una altra cosa que caldria garantir és que només l'usuari *root* (o dimonis associats a serveis) pugui escriure en el directori de registres `/var/log`. En cas contrari, qualsevol atacant podria falsificar la informació dels registres. Tot i així, si l'atacant aconsegueix accés a *root*, pot esborrar les pistes dels seus accessos (i acostuma a fer-ho).

9. Taller: anàlisi de la seguretat mitjançant eines

A continuació seguirem alguns dels processos descrits en els apartats previs sobre una distribució Debian (encara que es poden seguir de manera equivalent en una Fedora / Red Hat, la majoria de processos i mètodes són equivalents) per a configurar i auditar la seva seguretat (observarem diferents sortides de les ordres, depenent de la versió i sistema físic que utilitzem).

Podem començar amb algunes comprovacions locals. No entrem en els detalls, ja que moltes d'aquestes s'han comentat prèviament, i en alguns casos són procediments típics d'administració local (segons examinem en el material d'*Introducció a l'administració*):

- Comprovar que no hi hagi usuaris o grups problemàtics. Podem analitzar `/etc/passwd` i `/etc/group` cercant usuaris o grups que no corresponguin a usuaris de sistema, o siguin usuaris de sistema o associats a un determinat servei. En aquest últim cas, també és una bona manera de detectar serveis que no teníem identificats com en funcionament, i que possiblement procedeixin d'una instal·lació prèvia sense configuració o que han deixat de ser útils. Seria un bon moment per a desinstal·lar aquests serveis, si se sap que no s'utilitzaran, ja que probablement estiguin deficientment configurats en temes de seguretat.
- Comprovar que no existeixin usuaris amb *passwords* buits. Recordeu que si està funcionant `/etc/shadow`, el segon paràmetre en `/etc/passwd` hauria de ser `x`. Si simplement està buit: `::` o en `/etc/shadow`, si està buit sense disposar de *hash* (el camp amb `!` o `*` tampoc permet *login*). En aquestes dues últimes comprovacions també seria útil aprofitar l'ocasió per a inhabilitar comptes d'usuari que hàgim detectat com a no utilitzats, a més de verificar els últims comptes d'usuari afegits, per si detectem alguna incoherència amb la nostra activitat prèvia de gestió d'usuaris.
- Verificar també l'existència d'usuaris i grups amb Id 0 (associats a usuaris amb permisos de *root*).
- Verificar configuració de PAM, per a comprovar que no estigui alterada, especialment en aquells mòduls relacionats amb el procés de *login* i/o restriccions de *passwords* que puguem haver imposat.
- Verificar, amb ordres com `lastlog`, les últimes connexions dels usuaris per a detectar els últims usos interactius i des de quins llocs s'han fet.

- Comprovació de missatges de Syslog, Rsyslog o Systemd (journalctl), per a la detecció d'errors de sistema i autenticació.
- Verificació de l'existència d'arxius `Sticky Bit` i `suid/guid`. Podem comprovar que no s'hagin generat en el sistema arxius o directoris amb aquests permisos.
- Comprovació de signatures de binaris (i altres fitxers). Per exemple, si disposem de sumes prèvies, amb eines tipus `tripware` o `AIDE`.
- Verificació dels processos en execució. ordres com `ps`, `top`.
- Configuració de `sudoers` `/etc/sudoers`: quins usuaris disposen de permisos especials.
- Comprovació de treballs de cron: `/etc/cron.daily`, i altres períodes per a comprovar que no hagin estat introduïdes tasques no esperades.
- Comprovar amb el sistema de paquets si es disposa d'actualitzacions noves en el sistema, especialment correccions de seguretat.

Quant a comprovacions per a la seguretat en xarxa, detallarem alguns passos amb major detall, però una sèrie de tasques típiques en l'audició i configuració de seguretat serien:

- Verificar configuració de xarxa (*ifconfig*), rutes disponibles *route*.
- Verificació de *runlevel* actiu (*runlevel*), estat `/etc/inittab` o *target* equivalent en arrencada Systemd.
- Verificació de l'estat dels serveis de xarxa. Quins? Estat?
- Verificació de les connexions de xarxa. ordres `netstat`.
- Comprovar la configuració d'SSH (`/etc/ssh/sshd_config`) per a les opcions habituals de no *login* per *root*, SSH 2 activat, entre d'altres.
- Verificació de l'estat del *firewall* (`iptables`), comprovar regles, segueix actiu?
- Verificació d'alguns serveis problemàtics: Samba (està actiu i el necessitem?). Serveis d'NTP (ens permeten disposar del sistema sincronitzat).
- Verificació dels *logs* específics d'alguns servidors, normalment disponibles en `/var/log/servei`, tant si és un fitxer com un directori amb els *logs* d'accés i error pertinents, per exemple, de servidor web o bases de dades. Les eines comentades, com *logcheck* o *logwatch*, ens permeten també una millor detecció a través dels seus resums obtinguts dels *logs* dels serveis i del general del sistema.

Examinem ara amb més detall algunes d'aquestes fases per controlar i auditar la seguretat del nostre sistema. En aquest cas, ho fem a partir d'un sistema exemple amb una distribució Debian (en particular, una versió antiga que tenia alguns problemes de seguretat en la configuració per defecte).

Primer examinarem què ofereix, quant a serveis, la nostra màquina a la xarxa. Per a això, utilitzarem l'eina *nmap* com a escanejador de ports. Amb l'ordre (des de *root*):

```
nmap -sTU -O localhost
```

obtenim:

```
root@maquina:~# nmap -sUT -O localhost
starting nmap 5.21 ( nmap.org ) 11:31 CEST
Interesting ports on localhost (127.0.0.1):

(The 3079 ports scanned but not shown below are in state: closed)
Port      State Service
9/tcp     open  discard
9/udp     open  discard
13/tcp    open  daytime
22/tcp    open  ssh
25/tcp    open  smtp
37/tcp    open  time
37/udp    open  time
80/tcp    open  http
111/tcp   open  sunrpc
111/udp   open  sunrpc
113/tcp   open  auth
631/tcp   open  ipp
728/udp   open  unknown
731/udp   open  netviewdm3
734/tcp   open  unknown

Remote operating system guess: Linux kernel 2.6.X
Uptime 2.011 days
Nmap run completed 1 IP address (1 host up) scanned in 9.404 seconds
```

Podem observar que ens ha detectat un gran nombre de serveis oberts (depenent de la màquina, podria haver-n'hi més: Telnet, FTP, finger, etc.), tant en protocols tcp com udp. Alguns serveis, com *discard*, *daytime* o *time*, poden ser útils en alguna ocasió, però normalment no haurien d'estar oberts a xarxa, ja que es consideren insegurs. SMTP és el servei de reexpedició i encaminament del correu electrònic; si actuem com a *host* o servidor de correu, hauria d'estar actiu, però si només llegim i escrivim correu mitjançant comptes POP3 o IMAP, no té per què estar-ho.

Una altra manera de detectar serveis actius seria mitjançant la cerca de ports actius a les escoltes, detectant les connexions de xarxa actives; això pot fer-se amb l'ordre `netstat -lut`.

L'ordre `nmap` també pot aplicar-se amb el nom DNS o IP de la màquina; així veiem el que es veuria des de l'exterior (amb *localhost* veiem el que pot

veure la mateixa màquina) o, millor fins i tot, podríem utilitzar una màquina d'una xarxa externa (per exemple, un PC qualsevol connectat a Internet) per a examinar el que veurien de la nostra màquina des de l'exterior. En els primers casos, no estarem tallant les connexions pel tallafoc si aquest existís. Només des de l'exterior podrem comprovar si aquest efectivament tanca els ports esperats.

Anem ara a `/etc/inetd.conf` per desactivar aquests serveis (si han aparegut o no en l'escaneig previ dependrà de la distribució GNU/Linux i de la configuració prèvia d'aquests serveis, o pot ser que `inetd` ja no estigui actiu en les últimes distribucions). Busquem línies com:

```
discard stream tcp nowait root internal
smtp stream tcp nowait mail /usr/sbin/exim exim -bs
```

I els col·loquem una `#` al principi (només per a aquells serveis que vulguem desactivar i sapiguem què fan realment -consulteu les pàgines man- o es recomani la seva desactivació). Un altre cas de desactivació especialment recomanada seria la dels serveis d'FTP, Telnet, finger, etc. i fer servir `ssh` per a substituir-los.

Ara hem de reiniciar `inetd` perquè torni a llegir la configuració que hem canviat: `/etc/init.d/inetd restart`.

Tornem a `nmap`:

```
22/tcp open ssh
80/tcp open http
111/tcp open sunrpc
111/udp open sunrpc
113/tcp open auth
631/tcp open ipp
728/udp open unknown
734/tcp open unknown
```

D'allò que ens queda, tenim el servei `ssh`, que volem deixar actiu, i el servidor de web, que l'aturarem de moment:

```
/etc/init.d/apache2 stop
```

`ipp` és el servei d'impressió associat a CUPS. En administració local, vam veure que hi havia una interfície web de CUPS que es connectava al port 631. Si volem tenir una idea de a què es dedica un port determinat, podem mirar en `/etc/services`:


```
root@maquina:~# grep 631 /etc/services
ipp 631/tcp # Internet Printing Protocol
ipp 631/udp # Internet Printing Protocol
```

Si no estem actuant com a servidor d'impressió cap a l'exterior, hem d'anar a la configuració de CUPS i eliminar aquesta prestació (per exemple, col·locant un *listen* 127.0.0.1:631, perquè només escolti la màquina local) o limitar l'accés a les màquines permeses.

Ens apareixen també alguns ports com a desconeguts, en aquest cas el 728 i 734; això indica que nmap no ha pogut determinar quin servei està associat al port. Intentarem comprovar-ho directament. Per a això, executem sobre el sistema l'ordre *netstat*, que ofereix diferents estadístiques del sistema de xarxa, des de paquets enviats i rebuts, i errors, fins a allò que ens interessa, que són les connexions obertes i qui les fa servir. Intentem buscar qui està usant els ports desconeguts:

```
root@maquina:~# netstat -anp | grep 728
udp 0 0 0.0.0.0:728 0.0.0.0:* 552/rpc.statd
```

I si fem el mateix amb el 734, observem també que qui ha obert el port ha estat *rpc.statd*, que és un *daemon* associat a NFS (en aquest cas el sistema té un servidor NFS). Si fem aquest mateix procés amb els ports 111 que apareixien com a *sunrpc*, observarem que el *daemon* que hi ha darrere és *portmap*, que s'usa en el sistema de crides RPC. El sistema de crides RPC (*remote procedure call*) permet utilitzar el mecanisme de crides remotes entre dos processos que estan en diferents màquines. *portmap* és un *daemon* que s'encarrega de traduir les crides que li arriben pel port als números de serveis RPC interns que es tinguin, i és utilitzat per diferents servidors, com ara NFS, NIS i NIS+.

Els serveis RPC que s'ofereixen es poden veure amb l'ordre *rpcinfo*:

```
root@maquina:~# rpcinfo -p
programa vers proto port
100000 2 tcp 111 portmapper
100000 2 udp 111 portmapper
100024 1 udp 731 status
100024 1 tcp 734 status
391002 1 tcp 39797 sgi_fam
391002 2 tcp 39797 sgi_fam
```

on observem els serveis RPC amb alguns dels ports que ja s'havien detectat. Una altra ordre que pot resultar útil és *lsof*, que, entre altres funcions, permet relacionar ports amb els serveis que els han obert (per exemple: *lsof -i | grep 731*).

El *daemon* *portmap* és una mica crític amb la seguretat, ja que, en principi, no ofereix mecanismes d'autenticació del client, ja que se suposa que es deleguen en el servei (NFS, NIS, etc.). Per tant, *portmap* podria ser víctima d'intents de DOS/DDoS que podrien provocar errors en els serveis o fer-los caure. Normalment, protegiem *portmap* per mitjà d'algun tipus de *wrapper* i/o tallafoc. Si no utilitzem, i no tenim previst utilitzar, serveis NFS i NIS, el millor és desactivar completament *portmap*, traient-lo del *runlevel* en què s'activi. També podem parar-los momentàniament amb els *scripts* (en Debian):

```
/etc/init.d/nfs-common  
/etc/init.d/nfs-kernel-server  
/etc/init.d/portmap
```

I donar-los el paràmetre *stop* per a parar els serveis RPC (en aquest cas NFS). En tots aquests casos, només estem parant els serveis per a l'execució actual; si volem desactivar realment aquests serveis, hem d'anar al nivell d'execució concret (*runlevel*) i desactivar els serveis (com s'ha comentat en el mòdul), perquè si no, es reiniciaran en la següent arrencada del sistema.

A continuació, controlarem la seguretat sobre la base d'un *wrapper* senzill. Suposem que volem deixar pas a través d'SSH d'una màquina determinada, que denominem 1.2.3.4 (adreça IP). Tanquem *portmap* a l'exterior, ja que no tenim NIS, i de NFS tenim servidor però no estem servint res (podríem tancar-lo, però el deixarem per a futurs usos). Fem un *wrapper* (suposem que els *TCP wrappers* estan ja instal·lats, poden ser necessaris *tcpd* i *libwrap*), modificant els fitxers *hosts.deny* i *hosts.allow*. En */etc/hosts.deny*:

```
ALL : ALL : spawn (/usr/sbin/safe_finger -l @%h \  
| /usr/bin/mail -s "%c FAILED ACCESS TO %d!" root) &
```

Estem denegant tots els serveis (compte, aquells relacionats amb *inetd*) (primer *all*) a tots (*all*), i l'acció a prendre serà esbrinar qui ha demanat el servei (només funcionarà si la màquina suporta *finger*) i a quina màquina, i enviarem un correu al *root* que informi de l'intent. Podríem també escriure un fitxer de registre. Ara, en */etc/hosts.allow*:

```
sshd: 1.2.3.4
```

habilitem l'accés per a la màquina IP 1.2.3.4 en el servidor *sshd* (de l'*ssh*). Podem col·locar una llista de màquines o bé subxarxes que puguin utilitzar el servei (vegeu *man hosts.allow*). Recordeu que també tenim les ordres *tcpdchk*, per a comprovar que la configuració del *wrapper* sigui correcta, i

tcpdmatch, per a simular què passaria amb un determinat intent. Per exemple:

```
root@maquina:# tcpdmatch sshd 1.2.3.4

warning: sshd: no such process name in
/etc/inetd.conf client: hostname maquina.domini.es
client: address 1.2.3.4
server: process sshd
matched: /etc/hosts.allow line 13
access: granted
```

Ens comenta que seria concedit l'accés. Un detall és que ens explicita que sshd no està en el `inetd.conf` i, si ho verifiquem, veiem que així és: no s'activa pel servidor inetd, sinó per *daemon* propi (sshd) en el *runlevel* en què estiguem. A més (en Debian), aquest és un cas d'un dimoni que està compilat amb les biblioteques de *wrappers* incloses. En Debian hi ha diversos *daemons* que tenen aquest suport de *wrappers* en compilar-se amb les biblioteques pertinents com `rpc.statd`, `rpc.mountd`, `rpcbind` i `sshd`, entre d'altres. Això permet assegurar aquests *daemons* mitjançant *wrappers* pels fitxers *hosts* esmentats.

Una altra qüestió a verificar són les connexions actuals existents. Amb l'ordre `netstat -utp`, podem llistar les connexions tcp o udp establertes amb l'exterior, ja siguin entrants o sortints; així, en qualsevol moment podem detectar els clients connectats i a qui estem connectats. Una altra ordre important (de múltiples funcions) és `lsof`, que pot relacionar fitxers oberts amb processos o connexions per xarxa establerts mitjançant `lsof -i`, i així es poden detectar accessos indeguts a fitxers.

També podríem utilitzar un tallafoc per a processos similars (o bé com a mecanisme afegit). Començarem veient com estan les regles del tallafoc en aquest moment (ordre `iptables -L`):

```
root@aopcjj:# 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
```

És a dir, que el tallafoc no està col·locant cap restricció en aquest moment, i permet l'entrada, sortida i reexpedició de tots els paquets.

En aquest punt, podríem afegir un tallafoc que ens permetés una gestió més adequada de paquets que rebem i enviem, i que seria un control previ per a millorar la seguretat. Depenent de les nostres necessitats, establiríem les regles necessàries de manera semblant a aquelles que comentem en els exemples de tallafocs de la unitat.

En cas de col·locar actiu algun tallafoc, podem considerar si fer servir aquest mecanisme com a única garantia i treure els *wrappers*: podria fer-se, ja que els tallafocs (en aquest cas mitjançant iptables) ofereixen un mecanisme molt potent que ens permet seguir un paquet per tipus, per protocol i pel que està fent en el sistema. Un bon tallafoc podria ser més o menys suficient, però, per si de cas, més mesures de seguretat no vénen malament. I en cas que el tallafoc no estigués ben dissenyat i deixés escapar alguns paquets o *hosts*, el *wrapper* seria la mesura, en un nivell de servei, per a aturar els accessos no desitjats. Per a posar una metàfora que se sol utilitzar, si ens plantegéssim el nostre sistema com la defensa d'un castell medieval, el fossat i primeres muralles serien el tallafoc, i la segona muralla de contenció, els *wrappers*.

Les següents mesures ja podrien venir directament de cada servei (web, correu, impressió, etc.), amb les opcions de seguretat que ofereixi de manera addicional, mitjançant autenticació dels seus clients, mitjançant limitació dels accessos per perfils o ACL o simplement en oferir un subconjunt de funcionalitats requerides. Veurem en diferents mòduls orientats a serveis concrets algunes d'aquestes mesures en el nivell servei.

Com a opcions addicionals per a protegir el nostre sistema, podríem fer intervenir a algunes de les eines vistes en apartats anteriors, per exemple, per a mitjançant *Denyhosts* i *Fail2ban*, amb els procediments vistos en els subapartats 6.2. i 6.3., protegir les connexions SSH, i en el segon cas altres serveis com servidor web, correu, Samba, o altres que tinguem actius. Com a següent pas, podríem col·locar una eina de detecció de vulnerabilitats, com OpenVAS (vegeu el subapartat 6.1.), per a obtenir un informe de l'estat final del nostre servidor.

Resum

En aquest mòdul hem examinat els conceptes bàsics de seguretat aplicables als sistemes GNU/Linux, i hem identificat els possibles tipus d'atacs, tant locals com en sistemes en xarxa.

El coneixement del procés dels atacs ens permet dur a terme accions de seguretat activa mitjançant eines de detecció d'intrusions, i també de prevenció de possibles situacions problemàtiques.

Sistemes com SELinux ens permeten polítiques de seguretat, altament especificades, i ens ofereixen una àmplia sintaxi de permisos, controls i prevenció activa de la seguretat del sistema.

Cal examinar la seguretat tant des del punt de vista local del sistema, la qual cosa inclou la seguretat pel que fa a l'accés físic, com des del punt de vista dels sistemes en xarxa.

L'ús d'eines de seguretat en les diferents àrees de prevenció, detecció i actuació permet un control actiu de seguretat que ens pot evitar problemes més importants en els nostres sistemes.

També observem les limitacions de la seguretat dels sistemes informàtics i, en especial, les possibles sensacions d'una falsa seguretat total (difícil o impossible d'obtenir) que ens pot dur a una confiança cega, amb resultats pitjors que en el cas de no tenir-ne. La seguretat és un procés actiu que necessita un seguiment constant i participatiu per part de l'administrador de sistemes.

Activitats

1. Suposem que col·loquem un lloc web a la nostra màquina, per exemple amb Apache. El nostre lloc està pensat per a deu usuaris interns, però no controlem aquest nombre. Més endavant ens plantegem posar aquest sistema a Internet, ja que creiem que pot ser útil per als clients, i l'únic que fem és posar el sistema amb una IP pública a Internet. Quin tipus d'atacs podria patir aquest sistema?
2. Com podem detectar els fitxers amb SUID en el nostre sistema? Quines ordres caldran? I els directoris amb SUID o SGID? Per què cal, per exemple, que `/usr/bin/passwd` tingui bit de SUID?
3. Els fitxers `.rhosts`, tal com hem vist, són un perill important per a la seguretat. Podríem utilitzar algun mètode automàtic que en comprovés periòdicament l'existència? Com?
4. Com crearíem un ambient chroot per a `/bin/bash`? Descriviu els passos necessaris.
5. Suposem que volem deshabilitar un servei del qual sabem segur que el controla *l'script* `/etc/init.d/servei`: volem desactivar-lo en tots els nivells d'execució en què es presenta. Com trobem aquests nivells d'execució?, (per exemple, buscant enllaços a *l'script*). Com ho faríem en distribucions que suporten Systemd?
6. Examineu els serveis en actiu de la vostra màquina. Són tots necessaris? Com caldria protegir-los o desactivar-los?
7. Practiqueu l'ús d'algunes de les eines de seguretat que hem descrit (nmap, chkrootkit, wireshark, OpenVas, etc.).
8. Quines regles iptables serien necessàries per a una màquina amb la qual només vulguem accés per SSH des d'unes màquines habituals concretes?
9. I si volem només un accés al servidor web?

Bibliografia

- [Aus] **CERT Australia.** *Australian CERT.*
<<http://www.auscert.org.au>>
- [Bur02] **Burgiss, H.** (2002). *Security QuickStart HOWTO for Linux.* The Linux Documentation Project.
- [Cera] **CERT.** *CERT site.*
<<http://www.cert.org>>
- [Cerb] **CERT.** *CERT Vulnerability Database.*
<<http://www.kb.cert.org/vuls>>
- [Deb] **Debian.** *Lloc de seguretat de Debian.*
<<http://www.debian.org/security>>
- [Fbi] **Federal Bureau of Intelligence.** *Brigada del FBI para cibercriminals.*
<<http://www.fbi.gov/about-us/investigate/cyber/cyber>>
- [Fen02] **Kevin Fenzi.** *Linux security HOWTO.* The Linux Documentation Project.
- [Fri02] **Frisch, A.** (2002). *Essential System Administration* (3a. ed.). O'Reilly.
- [Gre] **Grennan, M.** *Firewall and Proxy Server HOWTO.* The Linux Documentation Project.
- [Hat08] **Hatch, B.** (2008). *Hacking Linux Exposed* (3a. ed.) McGraw-Hill.
- [Hatb] **Red Hat.** *Red Hat RHEL 7 Security Guide. 2014.*
<https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Security_Guide/index.html>
- [Hatc] **Red Hat.** *Lloc de seguretat de Red Hat, bases de dades de vulnerabilitats.*
<<https://access.redhat.com/security/cve/>>
- [Hatd] **Red Hat.** *Utilització de signatures GPG en Red Hat.*
<https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html-single/System_Administrators_Guide/#s1-check-rpm-sig>
- [Her13] **Hertzog, R; and Mas, R.** (2013). *El libro del administrador de Debian.*
<<http://debian-handbook.info/browse/es-ES/stable/index.html>>
- [Him01] **Himanen, P.** (2001). *La ética del hacker y el espíritu de la era de la información.* Destino.
- [Incb] **Internet Storm Center** (2014). *Análisis de Vulnerabilidades, y algunos incidentes.*
<<https://isc.sans.edu/>>
- [Ins] **Insecure.org** (1998). *Vulnerabilidades i exploits.*
<<http://www.insecure.org/sploits.html>>
- [Insa] **Insecure.org** (2014). *Insecure.org site.*
<<http://www.insecure.org>>
- [Insb] **Insecure.org** (2003). *Nmap Home site.*
<<http://nmap.org/index.html>>
- [Line] **Linuxsecurity.com** (2002). *Linux Security Reference Card.*
<<http://www.linuxsecurity.com/docs/QuickRefCard.pdf>>
- [Mor03] **Morill, D.** (2003). *Configuración de sistemas Linux.* Anaya Multimedia.
- [Mou02] **Mourani, G.** (2002). *Securing and Optimizing Linux: The Ultimate Solution. v2.0.* The Linux Documentation Project.
- [Nes] **Nessus.org.** *Nessus.*
<<http://www.nessus.org>>
- [Net] **Netfilter.org.** *Projecte Netfilter/IPTables.*
<<http://www.netfilter.org>>
- [Neu] **Neufeld, C.** *Setting Up Your New Domain MiniHOWTO.* The Linux Documentation Project.

- [Nsaa] **NSA**. *NIST site*.
<<http://csrc.nist.gov>>
- [Nsab] **NSA** (2009). *Security Enhanced Linux*.
<<http://www.nsa.gov/research/selinux/index.shtml>>
- [Opv] **OpenVas**. *OpenVas vulnerability Scanner*.
<<http://openvas.org>>
- [Pen] **Fernández-Sanguino Peña, J.** (2013). *Securing Debian Manual*.
<<http://www.debian.org/doc/manuals/securing-debian-howto/>>
- [Ray01] **Ray, John** (2011). *Maximum Linux Security: A Hacker's Guide to Protecting*. 2nd Edition, Sams.
- [San] **Sans** (2014). *Top20 de controls crítics de seguretat*.
<<http://www.sans.org/critical-security-controls/>>
- [Sei] **Seifried, K.** (2002). *Securing Linux, Step by Step*.
<<http://seifried.org/security/os/linux/20020324-securing-linux-step-by-step.html>>
- [Sno] **Snort.org**. *Snort*.
<<http://www.snort.org>>
- [Usa] **The United States Department of Justice**. *Divisió del Departament de Justícia dels Estats Units per al ciberdelicte*.
<<http://www.usdoj.gov/criminal/cybercrime/>>

Sobre aquestes fonts de referència i informació

- [Deb] [Hatc] Alguns llocs de seguretat de les distribucions.
- [Pen] Imprescindible per a Debian, molt bona opció per a seguir pas a pas la configuració de seguretat; [Hatb] seria equivalent per a Fedora / Red Hat.
- [Mou02] Excel·lent referència de seguretat per a Red Hat (aplicable també a Debian).
- [Hat08] Llibres sobre seguretat en GNU/Linux, que cobreixen un gran nombre d'aspectes i tècniques.
- [Line] Petita guia (2 pàgines) de seguretat.
- [Sei] Guia pas a pas d'identificació dels punts clau que cal verificar i els problemes que puguin sorgir.
- [Net] Projecte Netfilter/iptables i nftables.
- [Ian] Una llista de ports TCP/IP.
- [Proa] [Sno] [Insb] [Nes] Algunes de les eines de seguretat més utilitzades.
- [NSAb] Versió de Linux per a la seguretat, produïda per l'NSA. Referència per a SELinux.
- [CERa][Aus][Insa][Incb] [NSAa] Llocs d'organismes de seguretat.
- [CERb][Ins][San] Vulnerabilitats i *exploits* dels diversos sistemes operatius.
- [NSAa][FBI][USA] Alguns organismes governamentals de ciberdelictes als Estats Units.

Sintonització, optimització i alta disponibilitat

Remo Suppi Boldrito

PID_00212466

Índex

Introducció	5
Objectius	7
1. Sintonització, optimització i alta disponibilitat	9
1.1. Aspectes bàsics	9
1.1.1. Monitoratge sobre UNIX System V	10
1.1.2. Optimització del sistema	17
1.1.3. Optimitzacions de caràcter general	20
1.1.4. Configuracions complementàries	21
1.1.5. Resum d'accions per a millorar un sistema	25
1.2. Monitoratge.....	27
1.2.1. Munin	27
1.2.2. Monit	29
1.2.3. SNMP + MRTG	30
1.2.4. Nagios	33
1.2.5. Ganglia	35
1.2.6. Altres eines	36
1.3. Alta disponibilitat en Linux (<i>High-Availability Linux</i>)	37
1.3.1. Guia breu d'instal·lació de Heartbeat i Pacemaker (Debian)	38
1.3.2. DRBD	42
1.3.3. DRBD + Heartbeat com a NFS d'alta disponibilitat ...	44
Activitats	47
Bibliografia	48

Introducció

Un aspecte fonamental, una vegada que el sistema està instal·lat, és la configuració i adaptació del sistema a les necessitats de l'usuari i que les prestacions del sistema siguin el més adequades possible a les necessitats que ha de cobrir. GNU/Linux és un sistema operatiu eficient que permet un grau de configuració excel·lent i una optimització molt delicada d'acord amb les necessitats de l'usuari. És per això que, una vegada feta una instal·lació (o en alguns casos, una actualització), han de fer-se determinades configuracions vitals al sistema. Si bé el sistema “funciona”, és necessari efectuar alguns canvis (adaptació a l'entorn o sintonització) per a permetre que estiguin cobertes totes les necessitats de l'usuari i dels serveis que presta la màquina. Aquesta sintonització dependrà d'on es trobi funcionant la màquina i en alguns casos es farà per a millorar el rendiment del sistema, mentre que en uns altres (a més), per a qüestions de seguretat. Quan el sistema està en funcionament, és necessari monitorar-lo per a veure el seu comportament i actuar en conseqüència. Si bé és un aspecte fonamental, la sintonització d'un sistema operatiu moltes vegades es relega a l'opinió d'experts o gurus de la informàtica; però coneixent els paràmetres que afecten el rendiment, és possible arribar a les bones solucions fent un procés cíclic d'anàlisi, canvi de configuració, monitoratge i ajustos.

D'altra banda, amb les necessitats de serveis actuals, els usuaris són molt exigents amb la qualitat de servei que s'obté i és per això que un administrador ha de preveure les incidències que puguin ocórrer en el sistema abans que ocorrin. Per a això, és necessari que l'administrador “vigili” de manera continuada determinats paràmetres de comportament dels sistemes que el puguin ajudar en la presa de decisions i actuar per endavant evitant que es produeixi l'error, ja que si això passés podria suposar la possibilitat que el sistema deixés de funcionar, amb les possibles conseqüències econòmiques en alguns casos, però gairebé sempre amb la deterioració de la imatge de l'empresa/institució (a ningú no li agrada -o no li hauria d'agradar- que els usuaris informin d'una fallada en els sistemes d'informació).

En resum, un sistema de monitoratge ha d'ajudar l'administrador a reduir el MTTR (*Mean time to recovery*), que indica la mitjana de temps que un sistema triga a recuperar-se d'una fallada i que pot tenir un valor dins del contracte de qualitat de servei (normalment anomenat SLA, *Service Level Agreement*), per la qual cosa també pot tenir conseqüències econòmiques. Aquest valor de vegades es denomina “*mean time to replace/repair/recover/resolve*” en funció de quin sistema es tracti i de quin SLA s'hagi acordat, però en tots els casos estem parlant de la finestra de temps entre la qual es detecta un problema i les accions per a solucionar-lo. Amb el monitoratge, podrem tenir indicis d'aquests pro-

bles i executar les accions perquè no arribin a més i que si ocorren, tinguin el MTTR més baix possible.

En aquest mòdul es veuran les principals eines per a monitorar un sistema GNU/Linux, com són Munin, Monit, MRTG, Ganglia, Nagios, Cactis o Zabbix, i es donaran indicacions per a sintonitzar el sistema a partir de la informació obtinguda.

És important notar que si el MTTR és proper a zero, el sistema haurà de tenir redundància en els altres sistemes i és un aspecte important en l'actualitat per als servidors de sistemes de la informació, que es coneix com a *alta disponibilitat*.

Vegeu també

La seguretat s'estudia en el mòdul "Administració de seguretat".

L'alta disponibilitat (*high availability*) és un protocol de disseny del sistema, i la seva implementació associada assegura un cert grau absolut de continuïtat operacional durant períodes llargs de temps. El terme *disponibilitat* es refereix a l'habilitat de la comunitat d'usuaris per a accedir al sistema, enviar nous treballs, actualitzar o alterar treballs existents o recollir els resultats de treballs previs. Si un usuari no pot accedir al sistema es diu que està no disponible.

D'entre totes les eines que hi ha per a tractar aquests aspectes (Heartbeat, ldirectord per a LVS -Linux Virtual Server-, OpenSAF, Piranha, UltraMonkey, Pacemaker+Corosync, o Kimberlite (obs:EOL), etc.), en aquest mòdul analitzarem com es desenvolupa una arquitectura redundant en serveis utilitzant Heartbeat+DRBD.

Objectius

En els materials didàctics d'aquest mòdul, trobareu els continguts i les eines procedimentals per aconseguir els objectius següents:

- 1.** Analitzar i determinar les possibles pèrdues de prestacions d'un sistema.
- 2.** Solucionar problemes de sintonització del sistema.
- 3.** Instal·lar i analitzar les diferents eines de monitoratge i la seva integració per a resoldre els problemes d'eficiències/disponibilitat.
- 4.** Analitzar les eines que permeten tenir un sistema en alta disponibilitat.

1. Sintonització, optimització i alta disponibilitat

1.1. Aspectes bàsics

Abans de conèixer quines són les tècniques d'optimització, és necessari enumerar les causes que poden afectar les prestacions d'un sistema operatiu [31]. Entre aquestes, es poden esmentar:

1) **Colls d'ampolla en els recursos:** la conseqüència és que tot el sistema anirà més lent perquè hi ha recursos que no poden satisfer la demanda a la qual se'ls sotmet. El primer pas per a optimitzar el sistema és trobar aquests colls d'ampolla i determinar per què ocorren, coneixent les seves limitacions teòriques i pràctiques.

2) **Llei d'Amdahl:** segons aquesta llei, “hi ha un límit de quant es pot millorar en velocitat una cosa si només se'n optimitza una part”; és a dir, si es té un programa que utilitza el 10% de la CPU i s'optimitza reduint la utilització en un factor 2, el programa millorarà les prestacions (*speedup*) en un 5%, la qual cosa pot significar un tremend esforç no compensat pels resultats.

3) **Estimació de l'*speedup*:** és necessari estimar quant millorarà les prestacions el sistema per a evitar esforços i costos innecessaris. Es pot utilitzar la llei d'Amdahl per a valorar si és necessària una inversió, en temps o econòmica, en el sistema.

4) **Efecte bombolla:** sempre es té la sensació que, quan es troba la solució a un problema, en sorgeix un altre. Una manifestació d'aquest problema és que el sistema es mou constantment entre problemes de CPU i problemes d'entrada/sortida i viceversa.

5) **Temps de resposta enfront de quantitat de treball:** si es compta amb vint usuaris, millorar en la productivitat significarà que tots tindran més treball fet al mateix temps, però no millors respostes individualment; podria ser que el temps de resposta per a alguns fos millor que per a altres. Millorar el temps de resposta significa optimitzar el sistema perquè les tasques individuals triguin el menys possible.

6) **Psicologia de l'usuari:** dos paràmetres són fonamentals:

- a) l'usuari generalment estarà insatisfet quan es produeixin variacions en el temps de resposta; i
- b) l'usuari no detectarà millores en el temps d'execució menors del 20%.

7) Efecte prova: les mesures de monitoratge afecten les pròpies mesures. S'ha d'anar amb compte quan es fan les proves pels efectes col·laterals dels mateixos programes de mesura.

8) Importància de la mitjana i la variació: s'han de tenir en compte els resultats, ja que si s'obté una mitjana d'utilització de CPU del 50% quan ha estat utilitzada 100, 0, 0, 100, es podria arribar a conclusions errònies. És important veure la variació sobre la mitjana.

9) Coneixements bàsics sobre el maquinari del sistema que cal optimitzar: per a millorar una cosa és necessari “conèixer” si és susceptible de millorar. L'encarregat de l'optimització haurà de conèixer bàsicament el maquinari subjacent (CPU, memòries, busos, cau, entrada/sortida, discos, vídeo, etc.) i la seva interconnexió per a poder determinar on estan els problemes.

10) Coneixements bàsics sobre el sistema operatiu que cal optimitzar: de la mateixa manera que en el punt anterior, l'usuari haurà de conèixer aspectes mínims sobre el sistema operatiu que pretén optimitzar, entre els quals s'inclouen conceptes com processos i fils o *threads* (creació, execució, estats, prioritats, terminació), crides al sistema, *buffers* de cau, sistema d'arxius, administració de memòria i memòria virtual (paginació, *swap*) i taules del nucli (*kernel*).

1.1.1. Monitoratge sobre UNIX System V

El directori `/proc` el veurem com un directori, però en realitat és un sistema d'arxius fictici anomenat *procfs* (i que es munta en temps de *boot* de la màquina), és a dir, no existeix sobre el disc i el nucli el crea en memòria. S'utilitza per a proveir d'informació sobre el sistema (originalment sobre processos, d'aquí el nom), informació que després serà utilitzada per *totes* les ordres que veurem a continuació. El `/proc` actua com a interfície en l'estructura de dades internes de nucli i pot ser utilitzat per a canviar certa informació del kernel en temps d'execució (`sysctl`). No hem de confondre **procfs** amb el **sysfs** ja que aquest últim exporta informació sobre els dispositius i els seus controladors des del model de dispositius del nucli cap a l'espai de l'usuari (i és utilitzat per algunes parts importants del sistema com l'*udev*, que és el que crea per compatibilitat el `/dev`) permetent obtenir paràmetres i configurar-ne algun (p. ex., saber la grandària d'un disc `cat /sys/block/sda/size` o quins dispositius tenim en `/sys/class`). Una vista d'aquest directori és:

bus	locks	cgroups	meminfo
cmdline	misc	consoles	modules
cpuinfo	mounts	crypto	mtrr
devices	net	diskstats	pagetypeinfo
dma	partitions	dri	sched_debug
driver	self	execdomains	slabinfo
fb	softirqs	filesystems	stat
fs	swaps	interrupts	sys
iomem	sysrq-trigger	ioports	sysvipc
irq	timer_list	kallsyms	timer_stats
kcore	tty	keys	uptime

key-users	version	kmsg	vmallocinfo
acpi	kpagecount	vmstat	asound
kpageflags	zoneinfo	buddyinfo	loadavg

A més d'una sèrie de directoris numèrics que corresponen a cadascun dels processos del sistema.

En el directori `/proc` existeixen un conjunt d'arxius i directoris amb diferent informació. A continuació, en veurem alguns dels més interessants*:

*Consulteu la pàgina del manual per a obtenir més informació.

`/proc/1`: un directori amb la informació del procés 1 (el número del directori és el PID del procés).
`/proc/cpuinfo`: informació sobre la CPU (tipus, marca, model, prestacions, etc.).
`/proc/devices`: llista de dispositius configurats en el nucli.
`/proc/dma`: canals de DMA utilitzats en aquest moment.
`/proc/filesystems`: sistemes d'arxius configurats en el nucli.
`/proc/interrupts`: mostra quines interrupcions estan en ús i quantes se n'han processat.
`/proc/ioports`: ídem amb els ports.
`/proc/kcore`: imatge de la memòria física del sistema.
`/proc/kmsg`: missatges generats pel nucli, que després són enviats a syslog.
`/proc/ksyms`: taula de símbols del nucli.
`/proc/loadavg`: càrrega del sistema.
`/proc/meminfo`: informació sobre la utilització de memòria.
`/proc/modulis`: mòduls carregats pel nucli.
`/proc/net`: informació sobre els protocols de xarxa.
`/proc/stat`: estadístiques sobre el sistema.
`/proc/uptime`: des de quan el sistema està funcionant.
`/proc/version`: versió del nucli.

Aquests arxius es construeixen de manera dinàmica cada vegada que es visualitza el contingut i el nucli del sistema operatiu els proveeix en temps real. És per això que es denomina *sistema d'arxius virtual* i el contingut dels arxius i directoris va canviant en forma dinàmica amb les dades actualitzades. D'aquesta manera, es pot considerar el `/proc/` com una interfície entre el nucli de Linux i l'usuari i és una forma sense ambigüitats i homogènia de presentar informació interna i pot ser utilitzada per a les diverses eines/ordres d'informació/sintonització/control que utilitzarem regularment. És interessant, per exemple, veure la sortida de l'ordre `mount` i el resultat de l'execució `more /proc/mounts` és totalment equivalent!

S'ha de tenir en compte que aquests arxius són visibles (text), però algunes vegades les dades estan "en brut" i són necessàries ordres per a interpretar-les, que seran les que veurem a continuació. Els sistemes compatibles UNIX SV utilitzen les ordres `sar` i `sadc` per a obtenir estadístiques del sistema. En Debian és `atsar` (i `atsadc`), que és totalment equivalent a les que hem esmentat i posseeix un conjunt de paràmetres que ens permeten obtenir informació de tots els comptadors i informació sense processar del `/proc`. Debian també inclou el paquet `sysstat` que conté les ordres `sar` (informació general de l'activitat del sistema), `iostat` (utilització CPU i d'E/S), `mpstat` (informes globals per processador), `pidstat` (estadístiques de processos) i `sadf` (mostra informació del `sar` en diversos formats). L'ordre `atsar` llegeix comptadors i estadístiques del fitxer `/proc` i els mostra per la sortida estàndard. La primera manera de cridar l'ordre és (executar-la com a `root` o agregar l'usuari a la categoria corresponent del `sudoers` per a executar amb el `sudo`):

Vegeu també

En els següents subapartats s'ensenyarà a obtenir i modificar la informació del nucli de Linux treballant amb el sistema d'arxius `/proc`.

```
atsar opcions t [n]n
```

On mostra l'activitat en n vegades cada t segons amb una capçalera que mostra els comptadors d'activitat (el valor per defecte de $n = 1$). La segona manera de cridar-la és:

```
atsar -opcions -s time -e time -i sec -f file -n day#
```

L'ordre extreu dades de l'arxiu especificat per `-f` (que per defecte és l'arxiu `/var/log/atsar/atsarxx`, essent `xx` el dia del mes) i que van ser prèviament guardades per `atsadc` (s'utilitza per a recollir les dades, desar-les i processar-les i en Debian és a `/usr/lib/atsar`). El paràmetre `-n` pot ser utilitzat per a indicar el dia del mes i `-s`, `-i` l'hora d'inici i final, respectivament. Per a activar `atsadc`, per exemple, es podria incloure en `/etc/cron.d/atsar` una línia com la següent:

```
@reboot root test -x /usr/lib/atsadc && /usr/lib/atsar/atsadc /var/log/atsar/atsa'date +%d'
10,20,30,40,50 * * * * root test -x /usr/lib/atsar/atsa1 && /usr/lib/atsar/atsa1
```

La primera línia crea l'arxiu després d'un reinici i la segona guarda les dades cada 10 minuts amb el *shell script* `atsa1`, que crida el `atsadc`. En `atsar` (o `sar`), les opcions s'utilitzen per a indicar quins comptadors cal mostrar i alguns són:

Opcions	Descripció
u	Utilització de CPU
d	Activitat de disc
l (i)	Nombre d'interrupcions
v	Utilització de taules en el nucli
y	Estadístiques d'utilització de ttys
p	Informació de paginació i activitat de <i>swap</i>
r	Memòria lliure i ocupació de <i>swap</i>
l (L)	Estadístiques de xarxa
L	Informació d'errors de xarxa
w	Estadístiques de connexions IP
t	Estadístiques de TCP
U	Estadístiques d'UDP
m	Estadístiques d'ICMP
N	Estadístiques de NFS
A	Totes les opcions

Entre `atsar` i `sar` només existeixen algunes diferències quant a la manera de mostrar les dades i `sar` inclou unes quantes opcions més (o diferents). A continuació, es veuran alguns exemples d'utilització de `sar` (exactament igual que amb `atsar`, només pot haver-hi alguna diferència en la visualització de les dades) i el significat de la informació que genera:

Utilització de CPU: `sar -u 4 5`

```
Linux  debian  2.6.26-2-686  #1 SMP Thu May 28 15:39:35 UTC 2009  i686  11/30/2010
05:32:54  cpu %usr    %nice    %sys %irq %softirq    %wait %idle          _cpu_
05:33:05  all   3      0      8    0      0      88    0
05:33:09  all   4      0     12    0      0     84    0
05:33:14  all  15      0     19    1      0     65    0
...
05:41:09  all   0      0      1    0      0      0   99
```

%usr i %sys mostren el percentatge de temps de CPU en el mode usuari amb nice=0 (normals) i en el mode nucli. idle indica el temps no utilitzat de CPU pels processos en estat d'espera (no inclou espera de disc). wait és el temps que la CPU ha estat lliure quan el sistema estava efectuant entrada o sortida (per exemple, de disc). irq i softirq és el temps que la CPU ha dedicat a gestionar les interrupcions, que és un mecanisme de sincronització entre allò que fa la CPU i els dispositius d'entrada i sortida. En el cas idle=99% significa que la CPU està ociosa, per la qual cosa no hi ha processos per a executar i la càrrega és baixa; si idle \simeq i el nombre de processos és elevat, hauria de pensar-se a optimitzar la CPU, ja que podria ser el coll d'ampolla del sistema. En l'exemple podem veure que hi ha poca utilització de CPU i molt ús d'entrada i sortida, per la qual cosa es pot verificar que en aquest cas la càrrega del sistema l'està generant el disc (per a l'exemple, s'havien obert 5 còpies del programa OpenOffice Writer).

Nombre d'interrupcions per segon: `sar -I 4 5`

```
Linux  debian  2.6.26-2-686  #1 SMP Thu May 28 15:39:35 UTC 2009  i686  11/30/2010
05:46:30  cpu iq00 iq01 iq05 iq08 iq09 iq10 iq11 iq12 iq14 iq15          _intr/s_
05:46:34  all   0    0    0    0    33    0    0  134    4    5
05:46:37  all   0    0    0    0   54    1    0  227   38   13
05:46:42  all   0    0    0    0   41    0    0  167   10    8
```

Mostra la informació de la freqüència d'interrupcions dels nivells actius que es troben en `/proc/interrupts`. Ens és útil per a veure si hi ha algun dispositiu que està interrompent constantment el treball de la CPU. Consultant aquest arxiu, veurem que en l'exemple les més actives són la 9 (acpi), 12 (teclat), 14-15 (ide) i molt poc la 10 (usb).

Memòria i swap: `sar -r 4 5`

```
Linux  debian  2.6.26-2-686  #1 SMP Thu May 28 15:39:35 UTC 2009  i686  11/30/2010
05:57:10  memtot memfree buffers    cached  slabmem    swptot swpfree  _mem_
05:57:17  1011M   317M   121M    350M    30M     729M   729M
05:57:21  1011M   306M   121M    351M    30M     729M   729M
05:57:25  1011M   300M   121M    351M    30M     729M   729M
```

En aquest cas, memtot indica la memòria total lliure i memfree, la memòria lliure. La resta d'indicadors és la memòria utilitzada en buffers, la utilitzada en cau (de dades), slabmem és la memòria dinàmica del nucli i swptot/free

és l'espai total/lliure de *swap*. És important tenir en compte que si `memfree` $\simeq 0$ (no hi ha espai), les pàgines dels processos aniran a parar al *swap*, on hi ha d'haver lloc tenint en compte que això permetrà l'execució, però tot anirà més lent. Això s'ha de contrastar amb l'ús de CPU. També s'ha de controlar que la grandària dels *buffers* sigui adequada i estigui en relació amb els processos que estan portant a terme operacions d'entrada i sortida. És també interessant l'ordre `free`, que permet veure la quantitat de memòria en una visió simplificada:

	total	used	free	shared	buffers	cached
Mem:	1036092	711940	324152	0	124256	359748
-/+ buffers/cache:		227936	808156			
Swap:	746980	0	746980			

Això indica que d'1 Gb gairebé les 3/4 parts de la memòria estan ocupades i que aproximadament 1/3 són de cau. A més, ens indica que el *swap* no s'està utilitzant per a res, per la qual cosa podem concloure que el sistema està bé. Si volguéssim més detalls hauríem d'utilitzar l'ordre `vmstat` (amb més detalls que la `sar -r`) per a analitzar què és el que està causant problemes o qui està consumint tanta memòria. A continuació, es mostra una sortida de `vmstat` 1 10*:

*Consulteu el manual per a obtenir una descripció de les columnes.

```
procs -----memory----- ---swap-- -----io----- -system-- ----cpu----
r  b   swpd   free   buff  cache   si   so   bi   bo   in   cs  us  sy  id  wa
1  1     0 324820 124256 359796   0   0   23   11   20  112   0   0  99   1
0  0     0 324696 124256 359816   0   0    0   88    4   96   1   1  98   0
0  0     0 324716 124256 359816   0   0    0    0  106  304   0   0 100   0
0  0     0 324716 124256 359816   0   0    0    0  150  475   1   2  97   0
...
```

Utilització de les taules del nucli: `sar -v 4 5`

```
Linux  debian  2.6.26-2-686  #1 SMP Thu May 28 15:39:35 UTC 2009  i686  11/30/2010
06:14:02  superb-sz inode-sz  file-sz  dquota-sz  flock-sz  _curmax_
06:14:06      0/0    32968/36    3616/101976    0/0        13/0
06:14:10      0/0    32968/36    3616/101976    0/0        13/0
06:14:13      0/0    32984/36    3616/101976    0/0        13/0
06:14:17      0/0    32984/36    3616/101976    0/0        13/0
06:14:22      0/0    33057/36    3680/101976    0/0        13/0
```

En aquest cas, `superb-sz` és el nombre actual-màxim de *superblocks* mantingut pel nucli per als sistemes d'arxius muntats; `inode-sz` és el nombre actual-màxim d'*incore-inodes* en el nucli necessari, que és d'un per disc com a mínim; `file-sz` és el nombre actual-màxim d'arxius oberts, `dquota-sz` és l'ocupació actual-màxima d'entrades de quotes (per a més informació, consulteu `man sar -o atsar`). Aquest monitoratge es pot completar amb l'ordre `ps -Af` (*process status*) i l'ordre `top`, que mostraran l'activitat i estat dels processos en el sistema. A continuació, es mostren dos exemples de totes dues ordres (només algunes línies):

```

debian:/proc# ps -Alw
F S      UID      PID  PPID  C  PRI  NI  ADDR  SZ  WCHAN  TTY          TIME CMD
4 S      0        1      0  0  80   0  -    525  -      ?          00:00:01 init
5 S      0        2      0  0  75  -5  -      0  -      ?          00:00:00 kthreadd
1 S      0        3      2  0 -40   -  -      0  -      ?          00:00:00 migration/0
...
5 S      1    1601      1  0  80   0  -    473  -      ?          00:00:00 portmap
5 S     102    1612      1  0  80   0  -    489  -      ?          00:00:00 rpc.statd
...
4 S     113    2049    2012  0  80   0  -   31939  -      ?          00:00:03 mysqld
...
4 S      0    2654    2650  0  80   0  -    6134  -     tty7      00:00:49 Xorg
1 S      0    2726      1  0  80   0  -    6369  -      ?          00:00:00 apache2
0 S      0    2746      1  0  80   0  -     441  -     tty1      00:00:00 getty
...

```

Alguns aspectes interessants per a veure són la dependència dels processos (PPID = procés pare) i, per exemple, que per a saber l'estat dels processos es pot executar amb `ps -Alw` i en la segona columna ens mostrarà com es troba cadascun dels processos. Aquests paràmetres reflecteixen el valor indicat en la variable del nucli per a aquest procés, els més importants dels quals des del punt de vista del monitoratge són: *F flags* (en aquest cas 1 és amb superprivilegis, 4 creat des de l'inici *daemon*); *S* és l'estat (D: no interrompible dormint entrada/sortida, R: executable o en cua, S: dormint, T: en traça o aturat, Z: mort en vida, 'zombi'); *PRI* és la prioritat; *NI* és *nice*; *TTY*, des d'on s'ha executat; *TIME*, el temps de CPU; *CMD*, el programa que s'ha executat i els seus paràmetres. Si es vol sortida amb actualització (configurable), es pot utilitzar l'ordre `top`, que mostra unes estadístiques generals (processos, estats, càrrega, etc.) i, després, informació de cadascuna similar a la `ps`, però s'actualitza cada 5 segons per defecte (en mode gràfic està `gnome-system-monitor`):

```

top - 15:09:08 up 21 min,  2 users,  load average: 0.16, 0.15, 0.12
Tasks: 184 total,  2 running, 182 sleeping,  0 stopped,  0 zombie
%Cpu(s):  0.3 us,  2.8 sy,  0.0 ni, 96.8 id,  0.1 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:  1509992 total,  846560 used,  663432 free,  117304 buffers
KiB Swap: 1087484 total,    0 used, 1087484 free,  374076 cached
  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
 4144 root        20   0  201m  36m 8448 S   9.6  2.5   0:39.35 Xorg
 4694 adminp    20   0  980m  64m  29m S   6.7  4.4   0:26.22 gnome-shell
 4730 adminp    20   0  363m  16m  10m S   2.3  1.1   0:04.04 gnome-terminal
 4221 root        20   0 69796 1776 1140 S   0.3  0.1   0:01.62 nmbd
 4655 adminp    20   0  571m  26m  13m S   0.3  1.8   0:01.00 gnome-settings-
 6287 root        20   0 15080 1520 1072 R   0.3  0.1   0:00.10 top
    1 root        20   0 10648  812  676 S   0.0  0.1   0:01.21 init
    2 root        20   0      0    0    0 S   0.0  0.0   0:00.00 kthreadd
    3 root        20   0      0    0    0 S   0.0  0.0   0:00.93 ksoftirqd/0

```

Una ordre interessant i que presenta la informació d'una altra manera que pot servir per a tenir una panoràmica de tot el sistema és `atop` (s'ha d'instal·lar `apt-get install atop`). A continuació, unes línies d'aquesta ordre ens mostren la seva potencialitat:


```

ATOP - SySDW          2014/06/28 10:55:42          -----          17m58s elapsed
PRC | sys      2m06s | user   9.10s | #proc   184 | #zombie   0 | #exit    0 |
CPU | sys        7% | user    1% | irq      1% | idle    388% | wait     3% |
CPL | avg1     0.03 | avg5    0.07 | avg15   0.10 | csw 1116790 | intr 619733 |
MEM | tot      1.4G | free  659.0M | cache 369.7M | buff 100.2M | slab  64.5M |
SWP | tot      1.0G | free   1.0G |          | vmcom   2.1G | vmlim  1.8G |
DSK |          sda | busy    3% | read  27461 | write   2710 | avio  1.03 ms |
NET | eth0       0% | pcki   278 | pcko   260 | si     2 Kbps | so    0 Kbps |
NET | lo         --- | pcki    12 | pcko    12 | si     0 Kbps | so    0 Kbps |

    *** system and process activity since boot ***
  PID  SYSCPU  USRCPU  VGROW  RGROW  RDDSK  WRDSK  ST  EXC  S  CPU  CMD          1/27
3865  67.00s   2.31s  199.6M 36676K 14196K   148K N- - S   7% Xorg
4505  40.21s   4.14s  980.6M 68848K 25396K    8K N- - R   4% gnome-shell
4543   4.14s   0.60s  427.4M 16048K  4756K   160K N- - S   0% gnome-terminal

```

També es poden fer servir les eines del paquet `sysstat` per a conèixer l'estat dels recursos, com per exemple `vmstat` (estadístiques de CPU, memòria i entrada/sortida), `iostat` (estadístiques de discos i CPU) i `uptime` (càrrega de CPU i estat general).

Un resum de les ordres més interessants és:

Ordre	Descripció
<code>atop, top</code>	Activitat dels processos
<code>arpwatch</code>	monitor d'Ethernet/FDDI
<code>bmon, bwm-ng, nload</code>	monitor de l'amplada de banda
<code>downtimed</code>	Monitor del temps de caiguda, fora de servei
<code>free</code>	Utilització de memòria
<code>iostat, iotop</code>	Activitat de disc i E/S
<code>ip monitor, rtmon, iptotal, iptraf</code>	Monitor de dispositius de xarxa
<code>mpstat</code>	Estadístiques del processador
<code>netstat</code>	Estadística de la xarxa
<code>nfswatch</code>	Monitor de NFS
<code>ps, pstree, god</code>	Mostra estat i característiques dels processos
<code>/proc</code>	Sistema d'arxius virtual
<code>sar, atsar</code>	Recull informació del sistema
<code>stat, iwatch</code>	Estadístiques del sistema d'arxius
<code>strace</code>	Esdeveniments de les crides als sistemes i senyals
<code>tcpdump, etherape, sniffit</code>	Abocament / monitor de paquets de xarxa
<code>uptime, w</code>	Càrrega mitjana del sistema i temps des de l'inici
<code>vmstat</code>	Estadístiques de l'ús de memòria
<code>gnome-system-monitor, gkrellm, xosview, xwatch</code>	Monitors gràfics del sistema
<code>xconsole</code>	Monitor de missatges en l'escriptori

També hi ha una sèrie de programes que mesuren les prestacions del sistema (*benchmark*) o d'una part com per exemple: `netperf`, `mbw` (xarxa i amplada de banda), `iozone` (E/S), `sysbench`, `globs`, `gtkperf`, `hpcc` (general), `bonie++` (disc). És important destacar que el *benchmark* `hpcc` inclou *High-Performance LINPACK* (HPL), *benchmark* que és l'utilitzat per a mesurar les prestacions i fer el rànquing de les màquines més potents del món (<http://www.top500.org/>).

1.1.2. Optimització del sistema

A continuació, veurem algunes recomanacions per a optimitzar el sistema en funció de les dades obtingudes.

1) Resoldre els problemes de memòria principal: s'ha de procurar que la memòria principal pugui acollir un percentatge elevat de processos en execució, ja que si no és així, el sistema operatiu podrà pàginar i anar al *swap*; però això significa que l'execució d'aquest procés es degradarà notablement. Si s'agrega memòria, el temps de resposta millorarà força. Per a això, s'ha de tenir en compte la grandària dels processos (*SIZE*) en estat *R* i agregar-hi la que usa el nucli. Les quantitats de memòria es poden obtenir amb l'ordre *free*, que ens mostrarà (o amb *dmesg*), per exemple (*total/used/free/buffers/cached*):

```
1036092/723324/312768/124396/367472,
```

que és equivalent a (en megaoctets) 1011/706/305/121/358 i on observem, en aquest cas, que només el 30% de la memòria està lliure i que en aquest moment no hi ha problemes, però una càrrega mínima del sistema pot significar un problema. Per aquest motiu, haurem d'analitzar si el sistema està limitat per la memòria (amb *atsar -r i -p* es veurà molta activitat de paginació).

Les solucions per a la memòria són òbvies: o s'incrementa la capacitat o es redueixen les necessitats. Pel cost actual de la memòria, és més adequat incrementar la seva grandària que emprar moltes hores per a guanyar un centenar de bytes en treure, ordenar o reduir requeriments dels processos en la seva execució. Reduir els requeriments pot fer-se reduint les taules del nucli, traient mòduls, limitant el nombre màxim d'usuaris, reduint els *buffers*, etc.; tot això degradarà el sistema (efecte bombolla) i les prestacions seran pitjors (en alguns casos, el sistema pot quedar totalment no operatiu).

Un altre aspecte que es pot reduir és la quantitat de memòria dels usuaris gràcies a l'eliminació de processos redundants i canviant la càrrega de treball. Per a això, s'hauran de monitorar els processos que estan dormint (zombis) i eliminar-los, o bé aquells que no progressen en la seva entrada/sortida (saber si són processos actius, quanta CPU han gastat i si els usuaris els estan esperant). Canviar la càrrega de treball és utilitzar planificació de cues perquè els processos que necessiten gran quantitat de memòria es puguin executar en hores de poca activitat (per exemple, a la nit, llançant-los amb l'ordre *at*).

2) Molta utilització de CPU: bàsicament ens la dona el temps *idle* (valors baixos). Amb *ps* o *top* s'ha d'analitzar quins processos són els que "devoren CPU" i prendre decisions, com ara posposar la seva execució, parar-los temporalment, canviar la seva prioritat (és la solució menys conflictiva de totes i per a això es pot utilitzar l'ordre *renice prioritat PID*), optimitzar el programa (per a la propera vegada) o canviar la CPU (o agregar-ne una altra). Com

ja s'ha esmentat, GNU/Linux utilitza el directori `/proc` per a mantenir totes les variables de configuració del nucli que poden ser analitzades i, en cert cas, “ajustades”, per a aconseguir prestacions diferents o millors.

Per a això, s'ha d'utilitzar l'ordre `sysctl -a` per a obtenir totes les variables del nucli i els seus valors en l'arxiu*. Altres ordres alternatives són `sysctl` i `sysctldump`, que permeten descarregar les variables en un arxiu i modificar-les, per a carregar-les novament en el `/proc` (l'ordre `sysctl` guarda la configuració en `/etc/sysctl.conf`). En aquest cas, per exemple, es podrien modificar (s'ha de procedir amb cura, perquè el nucli pot quedar fora de servei) les variables de la categoria `/proc/sys/vm` (memòria virtual) o `/proc/sys/kernel` (configuració del *core* del nucli).

*Consulteu el manual per a canviar els valors i l'arxiu de configuració `/etc/sysctl.conf`

En aquest mateix sentit, també (per a experts o desesperats) es pot canviar el temps màxim (*slice*) que l'administrador de CPU (*scheduler*) del sistema operatiu dedica a cada procés en forma circular (si bé és aconsellable utilitzar `renice` com a pràctica). Però en GNU/Linux, a diferència d'altres sistemes operatius, és un valor fix dins del codi, ja que està optimitzat per a diferents funcionalitats (però és possible tocar-lo). Es pot “jugar” (sota la seva responsabilitat) amb un conjunt de variables que permeten tocar el *time slice* d'assignació de CPU (`kernel-source-x.x.x/kernel/sched.c`).

3) Reduir el nombre de crides: una altra pràctica adequada per a millorar les prestacions és reduir el nombre de crides al sistema de major cost en temps de CPU. Aquestes crides són les invocades (generalment) pel `shell fork()` i `exec()`. Una configuració inadequada de la variable `PATH` amb el directori actual (indicat per `.`) pot tenir una relació desfavorable d'execució (perquè la funció `exec()` no guarda res en cau i perjudica aquesta execució). Per a això, sempre caldrà configurar la variable `PATH` amb el directori actual com a última ruta. Per exemple, en `$HOME/.bashrc` s'ha de fer `PATH=$PATH:.; export PATH` si el directori actual no és en el *path* o, si hi és, refer la variable `PATH` per a posar-lo com a última ruta.

S'ha de tenir en compte que una alta activitat d'interrupcions pot afectar les prestacions de la CPU en relació amb els processos que executa. Mitjançant `monitor` (`atsar -I`), es pot mirar quina és la relació d'interrupcions per segon i prendre decisions pel que fa als dispositius que les causen. Per exemple, canviar de mòdem per un altre de més intel·ligent o canviar l'estructura de comunicacions si detectem una activitat elevada sobre el port sèrie on es troba connectat.

4) Molta utilització de disc: després de la memòria, un temps de resposta baix pot estar provocat pel sistema de discos. En primer lloc, s'ha de verificar que es disposi de temps de CPU (per exemple, `idle > 20%`) i que el nombre d'entrades/sortides sigui elevat (per exemple, superior a 30 entrades/sortides) utilitzant `atsar -o` i `atsar -d`. Les solucions passen per:

- En un sistema multidisc, planificar on es trobaran els arxius més utilitzats per a equilibrar el trànsit cap a aquests (per exemple `/home` en un disc i

/usr en un altre) i que puguin utilitzar totes les capacitats d'entrada/sortida amb cau i concurrent de GNU/Linux (fins i tot, per exemple, planificar sobre quin *bus ide* es col·loquen). Comprovar després que hi ha un equilibri del trànsit amb `atsar -d` (o amb `iostat`). En situacions crítiques, es pot considerar la compra d'un sistema de discos RAID que fan aquest ajust de manera automàtica.

- Tenir en compte que s'obtenen millors prestacions sobre dos discos petits que sobre un de la grandària dels dos anteriors.
- En sistemes amb un únic disc, generalment es fan, des del punt de vista de l'espai, quatre particions de la següent manera (des de fora cap a dins): */*, *swap*, */usr*, */home*. Però això genera pèssimes respostes d'entrada/sortida perquè si, per exemple, un usuari compila des del seu directori */home/user* i el compilador es troba en */usr/bin*, el cap del disc es mourà al llarg de tota la seva longitud. En aquest cas, és millor unir les particions */usr* i */home* en una de sola (més gran), encara que pot representar alguns inconvenients quant a manteniment.
- Incrementar els *buffers* de cau d'entrada/sortida (podeu veure, per exemple, */proc/ide/hd...*).
- Si s'utilitza un `extfs`, es pot fer servir l'ordre `dumpe2fs -h /dev/hdx` per a obtenir informació sobre el disc i `tune2fs /dev/hdx` per a canviar alguns dels paràmetres configurables d'aquest.
- Òbviament, el canvi del disc per un de major velocitat (més rpm) sempre tindrà un impacte positiu en un sistema limitat per l'entrada/sortida de disc [31].

5) Millorar aspectes de TCP/IP: examinar la xarxa amb l'ordre `atsar` (o també amb `netstat -i` o amb `netstat -s | more`) per a analitzar si hi ha paquets fragmentats, errors, *drops*, desbordaments, etc. que puguin estar afectant les comunicacions i, amb això, el sistema (per exemple, en un servidor de NFS, NIS, FTP o web). Si es detecten problemes, s'ha d'analitzar la xarxa per a considerar les següents actuacions:

- Fragmentar la xarxa mitjançant elements actius que descartin paquets amb problemes o que no siguin per a màquines del segment.
- Planificar on estaran els servidors per a reduir el trànsit cap a aquests i els temps d'accés.
- Ajustar paràmetres del nucli (*/proc/sys/net/*). Per exemple, per a obtenir millores en el *throughput* hem d'executar la següent instrucció: `echo 600 > /proc/sys/net/core/netdev_max_backlog*`.

6) Altres accions sobre paràmetres del nucli: hi ha altres paràmetres sobre el nucli que és possible sintonitzar per a obtenir millors prestacions, si bé, considerant el que hem tractat anteriorment, s'ha d'anar amb compte, ja que podríem causar l'efecte contrari o inutilitzar el sistema. Consulteu en la distribució del codi font en el directori *kernel-source-2.x/Documentation/sysctl* alguns arxius com per exemple *vm.txt*, *fs.txt* i *kernel.txt*. */proc/sys/vm* controla la memòria virtual del sistema (*swap*) i permet que els processos que no entren en la memòria principal siguin acceptats pel sistema però en el dispositiu de *swap*, per la qual cosa, el programador no té límit per a la grandària del seu programa (òbviament ha de ser menor que el dispositiu de *swap*). Els paràmetres susceptibles de sintonitzar es poden canviar molt fàcilment amb `sysctl` (o també amb `gpowertweak`). */proc/sys/fs* conté paràmetres que poden ser ajustats de la interacció nucli-sistema de fitxers, tal com `file-max` (i exactament igual per a la resta d'arxius d'aquest directori).

7) Generar el nucli adequat a les nostres necessitats: L'optimització del nucli significa escollir els paràmetres de compilació d'acord amb les nostres necessitats. És molt important primer llegir l'arxiu *readme* que hi ha al directori */usr/src/linux*.

Una bona configuració del nucli permetrà que s'executi més ràpid, que es disposi de més memòria per als processos d'usuari i, a més, resultarà més estable. Hi ha dues formes de construir un nucli: **monolític** (millors prestacions) o **modular** (basat en mòduls, que tindrà millor portabilitat si disposem d'un sistema molt heterogeni i no es desitja compilar un nucli per a cadascun). Per a compilar el seu propi nucli i adaptar-lo al seu maquinari i necessitats, cada distribució té les seves regles (si bé el procediment és similar).

1.1.3. Optimitzacions de caràcter general

Hi ha una sèrie d'optimitzacions d'índole general que poden millorar les prestacions del sistema:

1) Biblioteques estàtiques o dinàmiques: quan es compila un programa, es pot fer amb una biblioteca estàtica (`libr.a`), el codi de funció de la qual s'inclou en l'executable, o amb una de dinàmica (`libr.so.xx.x`), on es carrega la biblioteca en el moment de l'execució. Si bé les primeres garanteixen codi portable i segur, consumeixen més memòria. El programador haurà de decidir quina és l'adequada per al seu programa incloent `-static` en les opcions del compilador (no posar-ho significa dinàmiques) o `-disable-shared`, quan s'utilitza l'ordre `configuri`. És recomanable utilitzar (gairebé totes les distribucions noves ho fan) la biblioteca estàndard `libc.a` i `libc.so` de versions 2.2.x o superiors (coneguda com a Libc6) que reemplaça a les an-

Enllaços d'interès

És interessant consultar els següents llibres/articles:
<http://www.redbooks.ibm.com/redpapers/pdfs/redp4285.pdf> sobre Linux Performance and Tuning Guidelines,
http://people.redhat.com/alikins/system_tuning.html sobre informació d'optimització de sistemes servidors Linux, i
<http://www.linuxjournal.com/article/2396> sobre *Performance Monitoring Tools for Linux*. El primer és un llibre electrònic obert de la sèrie RedBooks d'IBM molt ben organitzat i amb gran quantitat de detalls sobre la sintonització de sistemes Linux, els dos restants són articles que si bé tenen un cert temps, els conceptes/metodologia i alguns procediments continuen vigents.

teriors. En gcc 4.X per defecte s'utilitzen biblioteques dinàmiques, però es pot forçar (no recomanat) l'ús d'estàtiques fins i tot per a la `libc` (opcions `-static -static-libgcc` en contraposició amb aquelles que hi ha per defecte `-shared -shared-libgcc`).

2) Selecció del processador adequat: generar codi executable per a l'arquitectura sobre la qual correran les aplicacions. Alguns dels paràmetres més influents del compilador són:

- a) `-march` (per exemple, `-march=core2` per al suport de CPU Intel Core2 CPU 64-bit amb extensions MMX, SSE, SSE2, SSE3/SSSE3, o `-march=k8` per a CPU AMD K8 Core amb suport x86-64) fent simplement
`gcc -march=i686;`
- b) l'atribut d'optimització `-O1,2,3` (`-O3` generarà la versió més ràpida del programa, `gcc -O3 -march = i686`), i
- c) els atributs `-f` (consulteu la documentació per als diferents tipus).

3) Optimització del disc: en l'actualitat, la majoria d'ordinadors inclou disc UltraDMA (100) per defecte; no obstant això, en una gran quantitat de casos no estan optimitzats per a extreure les millors prestacions. Hi ha una eina (`hdparm`) que permet sintonitzar el nucli als paràmetres del disc tipus IDE i SATA (encara que aquests últims també disposen d'una utilitat específica anomenada `sdparm`). S'ha d'anar amb compte amb aquestes utilitats, sobretot en discos UltraDMA (cal verificar en el BIOS que els paràmetres per a suport per a DMA estan habilitats), ja que poden inutilitzar el disc. Consulteu les referències i la documentació ([4] i `man hdparm/sdparm`) sobre quines són (i el risc que comporten) les optimitzacions més importants, per exemple: `-c3`, `-d1`, `-X34`, `-X66`, `-X12`, `-X68`, `-mXX`, `-a16`, `-o1`, `-W1`, `-k1`, `-K1`. Cada opció significa una optimització i algunes són d'altíssim risc, per la qual cosa caldrà conèixer molt bé el disc. Per a consultar els paràmetres optimitzats, es podria utilitzar `hdparm -vT /dev/hdX` (on X és el disc optimitzat), i la crida a `hdparm` amb tots els paràmetres es pot posar en `/etc/init.d` per a carregar-la en el *boot*. Per a consultar la informació del disc es pot fer, per exemple, `hdparm -i /dev/sdb`

Paquets no-free

Recordeu que sobre Debian s'ha d'activar el repositori de paquets `no-free` per a instal·lar els paquets de documentació del compilador `gcc-doc` i `gcc-doc-base`.

1.1.4. Configuracions complementàries

Hi ha més configuracions complementàries des del punt de vista de la seguretat que de l'optimització, però són necessàries sobretot quan el sistema està connectat a una intranet o a Internet. Aquestes configuracions impliquen les següents accions [4]:

1) Impedir que es pugui arrencar un altre sistema operatiu: si algú té accés físic a la màquina, podria arrencar amb un altre sistema operatiu preconfigurat i modificar l'actual, per la qual cosa s'ha d'inhibir des del BIOS de l'ordinador

el *boot* per CD-ROM o USB i posar una contrasenya d'accés (recordeu la contrasenya del BIOS ja que, d'una altra manera, podria causar problemes quan es volgués canviar la configuració).

2) Configuració i xarxa: és recomanable desconnectar la xarxa sempre que es desitgi fer ajustos en el sistema. Es pot treure el cable o deshabilitar el dispositiu amb `/etc/init.d/networking stop` (start per a activar-lo de nou) o amb `ifdown eth0` (`ifup eth0` per a habilitar-lo) per a un dispositiu en concret.

3) Modificar els arxius de */etc/security*: d'acord amb les necessitats d'utilització i seguretat del sistema. En *access.conf* hi ha informació sobre qui pot fer un *login* al sistema; per exemple:

```
# Taula de control d'accés. Línies amb columna1=# és un comentari.
# L'ordre de les línies és important.
# Format: permission : users : origins
# Deshabilitar tots els logins excepte root sobre tty1.
-:ALL EXCEPT root:tty1
# User "root" permès connectar-se des d'aquestes adreces.
+ : root : 192.168.200.1 192.168.200.4 192.168.200.9
+ : root : 127.0.0.1
# O des de la xarxa
+ : root : 192.168.201.
# Impedeix l'accés excepte user1,2,3 però l'últim només des de consola.
-:ALL EXCEPT user1 user2 user3:console
```

També s'haurien, per exemple, de configurar els grups per a controlar com i a on poden accedir i també els límits màxims (*limits.conf*) per a establir els temps màxims d'utilització de CPU, E/S, etc. i així evitar atacs per denegació de servei (DoS).

4) Mantenir la seguretat de la contrasenya de *root*: utilitzar com a mínim 8 caràcters, amb un, almenys, en majúscules o algun caràcter que sigui no trivial, com "-", ".", ",", etc.; així mateix, és recomanable activar l'envelliment per a forçar a canviar-lo periòdicament, així com també limitar el nombre de vegades amb contrasenya incorrecta. També es pot canviar el paràmetre `min=x` de l'entrada en */etc/pam.d/passwd* per a indicar el nombre mínim de caràcters que s'utilitzaran en la contrasenya (`x` és el nombre de caràcters). Hem d'utilitzar algorismes com ara SHA512 per a la configuració de *passwd* (en Debian ve configurat per defecte, vegeu */etc/pam.d/common-password*).

5) No accedir al sistema com a *root*: si bé moltes distribucions ja incorporen un mecanisme d'aquest estil (per exemple, Ubuntu), es pot crear un compte com *sysadm* i treballar-hi. Si s'hi accedeix remotament, sempre s'haurà d'utilitzar *ssh* per a connectar-se al *sysadm* i, en cas de ser necessari, fer un `su -` per a treballar com a *root* o activar el *sudoers* per a treballar amb l'ordre *sudo* (consulteu la documentació per a les opcions de l'ordre i la seva edició).

6) Temps màxim d'inactivitat: inicialitzar la variable `TMOUT`, per exemple a 360 (valor expressat en segons), que serà el temps màxim d'inactivitat que esperarà el *shell* abans de bloquejar-se; es pot posar en els arxius de configuració

del *shell* (per exemple, */etc/profile*, *.profile*, *\$HOME/.bashrc*, etc.). En cas d'utilitzar entorns gràfics (KDE, Gnome, etc.), es pot activar l'estalvi de pantalles amb contrasenya, igual que el mode de suspensió o hibernació.

7) Configuració del NFS en forma restrictiva: en el */etc/exports*, exportar només el necessari, no utilitzar comodins (*wildcards*), permetre solament l'accés de lectura i no permetre l'accés d'escriptura per *root*, per exemple, amb */directori_exportat host.domain.com (ro, root_squash)*.

8) Evitar arrencades des del *bootloader* amb paràmetres: es pot iniciar el sistema com a *linux single*, la qual cosa arrencarà el sistema operatiu en mode d'usuari únic. Cal configurar el sistema perquè l'arrencada d'aquesta manera sempre sigui amb contrasenya. Per a això, en l'arxiu */etc/inittab* s'ha de verificar que existeix la línia *S:wait:/sbin/sulogin* i que té habilitat el */bin/sulogin*. Verificar que els arxius de configuració del *bootloader* (*/etc/lilo.conf* si tenim LiLo com a gestor d'arrencada, o */etc/grub.d* si treballlem amb Grub2) han de tenir els permisos adequats perquè ningú els pugui modificar excepte el *root*. Mitjançant els arxius de configuració de *boot* es permeten una sèrie d'opcions que és convenient considerar: *timeout* per a controlar el temps de *boot*; *restricted* per a evitar que es puguin inserir ordres en el moment del *boot* com *linux init = /bin/sh* i tenir accés com a *root* sense autorització; en aquest cas, ha d'acompanyar-se de *password=paraula-de-password*; si només es posa *password*, sol·licitarà la contrasenya per a carregar la imatge del nucli (consulteu els manuals de LiLo/Grub per a la sintaxi correcta).

Es pot provar el risc que això representa fent el següent (sempre que el Grub/Lilo no tingui contrasenya), que pot ser útil per a entrar en el sistema quan no es recordi la contrasenya d'usuari, però representa un gran perill de seguretat quan és un sistema i es té accés a la consola i el teclat:

- a) s'arrenca l'ordinador fins que es mostri el menú d'arrencada,
- b) se selecciona el nucli que es desitja arrencar i s'edita la línia pressionant la tecla *e* (*edit*),
- c) busquem la línia que comença per *kernel ...* i al final de la línia esborrem el paràmetre *ro* i introduïm *rw init=/bin/bash* (la qual cosa indica accés directe a la consola). Pressionem F10.
- d) Amb això s'arrencarà el sistema i passarem directament a mode *root*, i gràcies a això es podrà canviar la contrasenya (inclosa la de *root*), editar el fitxer */etc/passwd* o el */etc/shadow* o també crear un nou usuari i tot el que desitgem.

9) Control de la combinació *Ctrl-Alt-Delete*: per a evitar que s'apagui la màquina des del teclat, s'ha d'inserir un comentari (#) en la primera columna de la línia següent: *ca:12345:ctrlaltdel:/sbin/shutdown -t1 -a -r now* de l'arxiu */etc/inittab*. Els canvis s'activen amb l'ordre *telinit q*.

10) Evitar peticions de serveis no oferts: s'ha de fer el bloqueig de l'arxiu */etc/services*, per a no admetre serveis no previstos, per mitjà de *chattr +i /etc/services*.

11) Connexió del *root*: cal modificar l'arxiu */etc/securetty* que conté les TTY i VC (*virtual console*) en què es pot connectar el *root* deixant només una de cada,

per exemple, `ttty1` i `vc/1` i, si és necessari, cal connectar-se com a `sysadm` i fer un `su`.

12) Eliminar usuaris no utilitzats: s'han d'esborrar els usuaris o grups que no siguin necessaris, inclosos els que vénen per defecte (per exemple, `operator`, `shutdown`, `FTP`, `uucp`, `games`, etc.) i deixar només els necessaris (`root`, `bin`, `daemon`, `sync`, `nobody`, `sysadm`) i els que s'hagin creat amb la instal·lació de paquets o per ordres (el mateix amb `/etc/group`). Si el sistema és crític, podria considerar-se el bloqueig (`chattr +i file`) dels arxius `/etc/passwd`, `/etc/shadow`, `/etc/group`, `/etc/gshadow` per a evitar la seva modificació (compte amb aquesta acció, perquè no permetrà canviar posteriorment les contrasenyes).

13) Muntar les particions en forma restrictiva: utilitzar en `/etc/fstab` atributs per a les particions com ara `nosuid` (que impedeix suplantar l'usuari o grup sobre la partició), `nodev` (que no interpreta dispositius de caràcters o blocs sobre aquesta partició) i `noexec` (que no permet l'execució d'arxius sobre aquesta partició). P. ex.: `/tmp /tmp ext2 defaults,nosuid,noexec 0 0`. També és aconsellable muntar el `/boot` en una partició separada i amb atributs `ro`.

14) Proteccions diverses: es pot canviar a 700 les proteccions dels arxius de `/etc/init.d` (serveis del sistema) perquè només el `root` pugui modificar-los, arrancar-los o parar-los i modificar els arxius `/etc/issue` i `/etc/issue.net` perquè no donin informació (sistema operatiu, versió, etc.) quan algú es connecta per `telnet`, `ssh`, etc.

15) SUID i SGID: un usuari podrà executar com a propietari una ordre si té el bit `SUID` o `SGID` activat, la qual cosa es reflecteix com una 's' `SUID` (`-rwsr-xr-x`) i `SGID` (`-r-xr-sr-x`). Per tant, és necessari treure el bit (`chmod a-s file`) a les ordres que no el necessiten. Aquests arxius poden buscar-se amb `find / -type f -perm -4000 o -perm -2000 -print`. S'ha de procedir amb cura respecte als arxius en què es tregui el `SUID`-`GUID`, perquè l'ordre podria quedar inutilitzada.

16) Arxius sospitosos: cal buscar periòdicament arxius amb noms no usuals, ocults o sense un uid/gid vàlid, com `"..."` (tres punts), `".. "` (punt punt espai), `"..^G"` o equivalents. Per a això, caldrà utilitzar `find / -name=".*" -print | cat -v`, o si no, `find / -name "..." -print`.

Per a buscar uid/gid no vàlids, utilitzeu `find / -nouser` o utilitzeu també `-nogroup` (compte, perquè algunes instal·lacions es fan amb un usuari que després no està definit i que l'administrador ha de canviar).

17) Connexió sense contrasenya: no s'ha de permetre l'arxiu `.rhosts` en cap usuari, tret que sigui estrictament necessari (es recomana utilitzar `ssh` amb clau pública en lloc de mètodes basats en `.rhosts`).

18) X Display manager: per a indicar els `hosts` que es podran connectar a través de `XDM` i evitar que qualsevol `host` pugui tenir una pantalla de `login` es pot modificar l'arxiu `/etc/X11/xdm/Xaccess`.

1.1.5. Resum d'accions per a millorar un sistema

1) Observar l'estat del sistema i analitzar els processos que consumeixen molta CPU utilitzant, per exemple, l'ordre `ps auxS -H` (o l'ordre `top`) i mirant les columnes `%CPU`, `%MEM` i `TIME`; s'ha d'observar la jerarquia de processos i parar esment a com s'està utilitzant la CPU i la memòria i analitzar el temps d'execució dels processos per a trobar processos *zombis* mirant en la columna `STAT` aquells que tinguin l'identificador `Z` (els quals es podran eliminar sense problemes). També s'ha de prestar especial atenció a aquells que estiguin amb `D`, `S` (que estan fent entrada o sortida) i `W` (que estan utilitzant el *swap*). En aquests tres últims, utilitzeu l'`atsar` i `free` (`sar` o `vmstat`) per a verificar la càrrega d'entrada i sortida, ja que pot ser que aquests processos estiguin fent que les prestacions del sistema baixin notablement (generalment, per les seves necessitats; en aquest cas, no podrem fer gran cosa, però en altres casos pot ser que el codi no estigui optimitzat o ben escrit).

2) Analitzar l'estat de la memòria detalladament per a descobrir on s'està gastant la memòria. Recordeu que tots els processos que s'han d'executar han d'estar en memòria principal i, si no n'hi ha, el procés paginarà en *swap* però amb la consegüent pèrdua de prestacions, ja que ha d'anar al disc i portar zona de memòria principal. És vital que els processos més actius tinguin memòria principal i això es pot aconseguir canviant l'ordre d'execució o fent un canvi de prioritats (ordre `renice`). Per a observar l'estat de la memòria detalladament, utilitzeu el `vmstat 2` (o l'`atsar`), per exemple, i observeu les columnes `swpd`, que és la quantitat de memòria virtual (*swap*) utilitzada, `free`, la quantitat de memòria principal lliure (l'ocupada s'obté de la total menys la lliure) i `si/so`, la quantitat de memòria virtual en lectura o escriptura utilitzada. Si tenim un procés que utilitza gran quantitat de *swap* (`si/so`) aquest procés estarà gastant molt de temps en gestió, retardarà el conjunt i veurem que la CPU té, per exemple, valors d'utilització baixos. En aquest cas, s'haurien d'eliminar processos de la memòria per a fer lloc o ampliar la memòria RAM del sistema si és que no es poden treure els processos que hi ha, sempre que el procés sota estudi no sigui d'execució ocasional.

3) Considereu que `%CPU + %E/S + %Idle = 100%` per la qual cosa veiem que l'E/S (I/O en `vmstat`) també afecta un procés.

```
debian:/home/remo# vmstat 2
procs -----memory----- --swap-- ----io---- -system-- ----cpu----
r  b   swpd   free   buff  cache   si   so    bi   bo    in   cs us  sy id wa
...
0  0       0 623624 29400 231596    0    0  7184    0  393 1389 10   8 10 73
0  0       0 623596 29400 231612    0    0    0    0  416  800  0   2 98  0
1  0       0 622540 29408 231604    0    0    0  276  212  549  2   2 94  2
0  0       0 613544 29536 240620    0    0  4538    0  464 1597 10   8 29 54
0  0       0 612552 29560 240824    0    0   112    0  412  850  1   2 85 12
```

En aquest cas, podem observar que hi ha una utilització molt gran d'I/O (E/S) però 0 de *swap* i un alt valor de CPU tant en `wa` (*waiting*) com en `id` (*idle*), la qual cosa vol dir que la CPU està esperant que alguna cosa que està en I/S acabi (en aquest cas és l'execució d'unes quantes instàncies del LibreOffice, la qual

cosa significa lectura de disc i càrrega d'un executable a memòria principal). Si aquesta situació es repeteix o és constant, s'hauria d'analitzar com utilitzen la memòria els processos i en espera d'execució i com es pot reduir (per exemple, posant un disc més ràpid o amb més *buffer* d'E/S).

4) S'ha de tenir en compte que el %CPU està constituït per la suma de dos valors "us" (*user time*) i "sy" (*system time*). Aquests valors representen el temps emprat executant codi de l'usuari (*non-kernel code*) i el temps gastat executant codi del nucli, respectivament, i poden ser útils quan es desitja optimitzar el codi d'un programa amb la finalitat que consumeixi menys temps de CPU. Utilitzarem l'ordre `time`, que ens dóna el temps gastat en cada tipus de codi, fent, per exemple, `time find /usr`, de manera que ens dóna `real 1m41.010s, user 0m0.076s, sys 0m2.404s`; en canvi, si fem `time ls -R /usr` la sortida és `real 0m5.530s user 0m0.160s sys 0m0.068s`. Com veiem, per a l'obtenció d'informació equivalent (llista d'arxius i directoris) una ordre ha gastat 2,404 s en espai de nucli i l'altra 0,06 s, per la qual cosa és interessant analitzar quines ordres escollim per a fer el treball. Un altre aspecte interessant és que si executem, per exemple, `time find /var > /dev/null` (per a no veure la sortida) la primera vegada obtenim `real 0m23.900s, user 0m0.000s, sys 0m0.484s` però una segona vegada obtenim `real 0m0.074s, user 0m0.036s, sys 0m0.036s`. Què ha passat? El sistema ha emmagatzemat en les taules de cau la informació i les següents vegades ja no triga el mateix, sinó molt menys. Si es desitja utilitzar el `time` en el format avançat o estès, els usuaris que executin *bash* com *shell* hauran d'executar el `time` juntament amb el *path* on es trobi; per exemple `/usr/bin/time ls -R /usr`, per a obtenir els resultats desitjats (consulteu `man time` per a més informació).

5) És interessant veure quines optimitzacions podem generar en un codi amb modificacions simples. Per exemple, observem el codi desenvolupat per Bravo [3]:

```
#include <stdio.h>
#include <sys/time.h>
#include <time.h>
int main(void)
{int x=1, y=2, z=3; long iter1=0, iter2=0;
 struct timeval tv1, tv2;
 gettimeofday(&tv1, NULL);
 for(;;) {
     x=(x*3+y*7+z*9)%11;
     y=(x*9+y*11+z*3)%29;
     z=(x*17+y*13+z*11)%37;
     iter1++;
     if(iter1==1000000){ iter2++; iter1=0;}
     gettimeofday(&tv2, NULL);
     if(tv2.tv_sec==tv1.tv_sec+5 && tv2.tv_usec>=tv1.tv_usec || tv2.tv_sec>tv1.tv_sec+5)
     break;}
     printf("Iteracions: %ldM Resultat: %d %d %d\n", iter2, x, y, z);
     return 0;
 }
```

El resultat de l'execució és

```
time ./c:Iteracions: 22M real 0m5.001s, user 0m1.756s, sys 0m3.240s
```

on es pot observar que els 3,240 s'han aconseguit per a 22 milions d'iteracions. En què es gasten els 3,240 s? Doncs a calcular l'hora en cada iteració, ja que són múltiples les crides al nucli. Si millorem el codi i només calculem el `gettimeofday` cada milió d'iteracions, obtenim `Iteracions: 135M real 0m5.025s, user 0m4.968s, sys 0m0.056s` i veiem que es redueix notablement el temps `sys` i obtenim més temps per a executar el codi de l'usuari, per la qual cosa puja el nombre d'iteracions (135 milions), tenint en compte que hem passat de 22 milions d'execució de la funció `gettimeofday` a 135 milions de vegades. Quina n'és la conseqüència? Que la finalització d'execució d'aquest exemple s'obté per comparació de 5 s amb el temps absolut, per la qual cosa en calcular el temps absolut menys vegades s'obté certa "imprecisió" en determinar quan finalitza l'execució (`real 0m5.001s` en el primer cas, mentre que en el segon `0m5.025s`, una diferència de 24 mil·lèsimes). Una solució optimitzada per a aquest cas seria no fer servir aquest tipus de crides al sistema per a determinar quan ha de finalitzar un procés i buscar alternatives, per exemple, amb `alarm` i una crida a `signal`. [3]

1.2. Monitoratge

Un aspecte important en el funcionament 24x7 d'un sistema és que l'administrador s'ha d'anticipar als problemes i és per això que o bé està contínuament mirant el seu funcionament (la qual cosa és pràcticament impossible tot el temps) o bé disposa d'eines adequades que puguin prevenir la situació, generar alertes i advertir el responsable que "alguna cosa està passant" perquè aquest pugui fer amb antelació les accions correctives per a evitar la fallada, disfunció o situació de fora de servei del sistema o recurs. Les eines que compleixen aquesta funció s'emmarquen dins del grup d'eines de monitoratge i permeten també obtenir informació del sistema amb finalitats estadístiques, comptables o altres que l'usuari desitgi. Les eines més comunes permeten, mitjançant una interfície web, conèixer de manera remota els cinc factors principals (ús de CPU, memòria, E/S, xarxa, processos/servis) que donen indicis que "alguna cosa pot estar passant"; les més sofisticades generen alarmes per SMS per a advertir de la situació l'administrador. A continuació, es descriuran algunes de les eines més representatives (però no són les úniques): Munin, Monit, MRTG, Nagios, Ganglia, Zabbix i Cacti.

1.2.1. Munin

Munin [8] produeix gràfics sobre diferents paràmetres del servidor (load average, memory usage, CPU usage, MySQL throughput, eth0 traffic, etc.) sense excessives configuracions i presenta gràfics importants per a reconèixer on i què està generant problemes. Considerem que el nostre sistema es diu `sysdw.nteum.org` i que ja el tenim configurat amb aquest nom i amb el DocumentRoot d'Apache en `/var/www/`. Per a instal·lar Munin sobre Debian fem, per exemple, `apt-get install munin munin-node`. Després hem de configurar Munin (`/etc/munin/munin.conf`) amb:

```
dbdir /var/lib/munin
htmldir /var/www/munin
logdir /var/log/munin
rundir /var/run/munin
tmpldir /etc/munin/templates
[debian.nteum.org]
address 127.0.0.1
use_node_name yes
```

Després es crea el directori, es canvien els permisos i es reinicia el servei (en cas de no existir).

```
mkdir -p /var/www/munin
chown munin:munin /var/www/munin
/etc/init.d/munin-node restart
```

Finalment, Munin només està configurat per a connectar-se des del *localhost*; si ho desitgem fer des d'una altra màquina, aleshores hem de canviar en */etc/apache2/conf.d/munin* (que és un enllaç a */etc/munin/apache.conf*) comentant la línia `#Allow from localhost 127.0.0.0/8 ::1` per a `Allow from all` i reiniciar Apache (`service apache2 restart`).

Després d'uns minuts, es podran veure els primers resultats en l'adreça web `http://localhost/munin`, en el navegador (o també el domini que tenim assignat en */etc/hosts*, per exemple, en el nostre cas *sysdw.nteum.org*). Si es vol mantenir la privadesa de les gràfiques, n'hi ha prou de posar una contrasenya per a l'accés amb Apache al directori. Per exemple, es posa en el directori */var/www/munin/* l'arxiu `.htaccess` amb el següent contingut:

```
AuthType Basic
AuthName "Members Only"
AuthUserFile /etc/munin/htpasswd
require valid-user
```

Després, s'ha de crear l'arxiu de contrasenya en */etc/munin/htpasswd* amb l'ordre (com a *root*): `htpasswd -c /etc/munin/htpasswd admin`. Quan ens connectem al `http://localhost/munin/`, ens demanarà l'usuari (admin) i la contrasenya que hem introduït després de l'ordre anterior.

Munin ve amb un conjunt de *plugins* instal·lats però fàcilment se'n poden habilitar uns altres fent, per exemple per a monitorar MySQL:

```
cd /etc/munin/plugins
ln -s /usr/share/munin/plugins/mysql_ mysql_
ln -s /usr/share/munin/plugins/mysql_bytes mysql_bytes
ln -s /usr/share/munin/plugins/mysql_innodb mysql_innodb
ln -s /usr/share/munin/plugins/mysql_isam_space_ mysql_isam_space_
ln -s /usr/share/munin/plugins/mysql_queries mysql_queries
ln -s /usr/share/munin/plugins/mysql_slowqueries mysql_slowqueries
ln -s /usr/share/munin/plugins/mysql_threads mysql_threads
```

1.2.2. Monit

Monit [7] permet configurar i verificar la disponibilitat de serveis com ara Apache, MySQL o Postfix i fa diferents accions, com per exemple reactivar-los si no estan presents. Per a instal·lar Monit, fem `apt-get install monit` i editem `/etc/monit/monitrc`. L'arxiu per defecte inclou un conjunt d'exemples, però s'haurà de consultar la documentació per a obtenir més informació*. A continuació, presentem un exemple de configuració típic sobre alguns serveis `/etc/monit/monitrc` [32]:

*<http://mmonit.com/monit>

```
# Monit control file example: /etc/monit/monitrc
# Només es mostren les línies canviades.
set daemon 120 # Poll at 2-minute intervals
set logfile /var/log/monit.log
set alert adminp@sysdw.nteum.org
# S'utilitza el servidor intern que disposa monit per a controlar apache2 també
set httpd port 2812 and
    use address localhost # only accept connection from localhost
    allow admin:monit # require user 'admin' with password 'monit'
# Exemples de monitors
check process sshd with pidfile /var/run/sshd.pid
    start program "/etc/init.d/ssh start"
    stop program "/etc/init.d/ssh stop"
    if failed port 22 protocol ssh then restart
    if 5 restarts within 5 cycles then timeout
check process mysql with pidfile /var/run/mysqld/mysqld.pid
    group database
    start program = "/etc/init.d/mysql start"
    stop program = "/etc/init.d/mysql stop"
    if failed host 127.0.0.1 port 3306 then restart
    if 5 restarts within 5 cycles then timeout
check process apache with pidfile /var/run/apache2.pid
    group www-data
    start program = "/etc/init.d/apache2 start"
    stop program = "/etc/init.d/apache2 stop"
    if failed host sysdw.nteum.org port 80 protocol http
        and request "/monit/token" then restart
    if cpu is greater than 60% for 2 cycles then alert
    if cpu > 80% for 5 cycles then restart
check process ntpd with pidfile /var/run/ntpd.pid
    start program = "/etc/init.d/ntp start"
    stop program = "/etc/init.d/ntp stop"
    if failed host 127.0.0.1 port 123 type udp then restart
    if 5 restarts within 5 cycles then timeout
```

Enllaç d'interès

Consulteu el manual per a obtenir més detalls a <http://mmonit.com/monit>.

Per al monitoratge d'Apache, hem indicat que verifiqui un fitxer que haurà d'estar en `/var/www/monit/token`, per la qual cosa s'haurà de crear amb:

```
mkdir /var/www/monit; echo "hello" > /var/www/monit/token
```

Per a verificar que la sintaxi és correcta, executem `monit -t` i per a engegar-lo executem `monit`. A partir d'aquest moment, es pot consultar en l'adreça i port seleccionat en l'arxiu `/etc/monit/monitrc` (en el nostre cas en l'adreça `http://localhost:2812/`), que ens demanarà l'usuari i la contrasenya també introduïts en el mateix arxiu (admin i monit en el nostre cas). Es pot parar i arrencar el servei amb `service monit restart` (verifiquem el valor de la variable `START` en `/etc/default/monit`).

1.2.3. SNMP + MRTG

El **MRTG** (*Multi-Router Traffic Grapher*) [9] va ser creat per a mostrar informació gràfica sobre dades de xarxa, com es veurà en l'exemple que mostrarem a continuació, per a monitorar la xarxa Ethernet, però es poden usar altres dades per a visualitzar el comportament i per a generar les estadístiques de càrrega (*load average*) del servidor. En primer lloc, instal·larem SNMP o Protocol Simple d'Administració de Xarxa (*Simple Network Management Protocol*), que és un protocol de la capa d'aplicació que facilita l'obtenció d'informació entre dispositius de xarxa (p. ex., *routers*, *switches*, servidors, estacions de treball, impresores, etc.) i que permet als administradors supervisar el funcionament de la xarxa i buscar/resoldre els seus problemes. Per a això fem `apt-get install snmp snmpd`. Les variables accessibles a través de SNMP estan organitzades en jerarquies metadades (tipus, descripció, etc.) i emmagatzemades en unes taules anomenades *Management Information Bases* (MIB). Per a instal·lar-les, hem d'agregar primer el repositori de Debian en non-free i després descarregar el paquet:

```
Agreguem en /etc/apt/sources.list
deb http://ftp.debian.org/debian wheezy main contrib non-free
```

```
Després actualitzem els repositoris i instal·lem el paquet
apt-get update
apt-get install snmp-mibs-downloader
download-mibs
```

Després hem de configurar el servei `snmpd` i per a això editem la configuració `/etc/snmp/snmpd.conf` (només hem deixat les línies més importants per a canviar):

```
agentAddress udp:127.0.0.1:161
rocommunity public
com2sec local localhost public
group MyRWGroup v1 local
group MyRWGroup v2c local
group MyRWGroup usm local
view all included .1 80
access MyRWGroup ""any noauth exact all all none
com2sec notConfigUser default mrtg
group notConfigGroup v1 notConfigUser
group notConfigGroup v2c notConfigUser
view systemview included .1.3.6.1.2.1.1
view systemview included .1.3.6.1.2.1.25.1.1
view systemview included .1 80
access notConfigGroup ""any noauth exact systemview none none
syslocation BCN
syscontact Adminp <adminp@sysdw.nteum.org>
```

A continuació, en l'arxiu `/etc/default/snmpd` hem modificat la línia `export MIBS=/usr/share/mibs` per a indicar-li on estaven les MIB i es reinicia el servei (`/etc/init.d/snmpd restart`). Podem interrogar al servidor `snmpd` utilitzant l'ordre `snmpwalk`, per exemple:

```
snmpwalk -v1 -c public localhost Donarà una llarga llista d'informació
snmpwalk -v 2c -c public localhost Igual que l'anterior
O preguntar-li per una variable específica de la MIB:
snmpwalk -v1 -c mrtg localhost IP-MIB::ipAdEntIfIndex
IP-MIB::ipAdEntIfIndex.127.0.0.1 = INTEGER: 1
IP-MIB::ipAdEntIfIndex.158.109.65.67 = INTEGER: 2
```

Amb això, ja podem instal·lar MRTG fent

```
apt-get install mrtg mrtg-contrib mrtgutils
```

Després generem la configuració amb

```
cfgmaker public@localhost > /etc/mrtg.cfg
```

i hem de crear el directori i canviar les proteccions per al grup d'Apache:
`/var/www/mrtg; chown www-data:www.data /var/www/mrtg`. Finalment,
 haurem de crear l'`index.html` amb:

```
indexmaker -title=L·Localhost--output /var/www/mrtg/index.html
/etc/mrtg.cfg.
```

Abans d'executar `mrtg` hi ha un error en Debian Wheezy i IPv6 que podem corregir amb [10]:

Quan executem `LANG=C /usr/bin/mrtg /etc/mrtg.cfg` obtenim l'error:

```
Subroutine SNMP_Session::pack_sockaddr_in6 redefined ...
Solució: editar l'arxiu /usr/share/perl5/Snmp_session.pm
En la línia 149:
On diu: import Socket6;
Reemplaçar per: Socket6->import(qw(inet_pton getaddrinfo));
En la línia 609:
on diu: import Socket6;
Reemplaçar per: Socket6->import(qw(inet_pton getaddrinfo));
```

Per a executar `mrtg` inicialment i verificar si tot està bé, hauríem de fer:
`env LANG=C /usr/bin/mrtg /etc/mrtg.cfg` (i repetint-la una sèrie de vegades). Podrem visualitzar l'activitat de xarxa accedint des del navegador a l'URL: <http://sysdw.nteum.org/mrtg/> i podrem veure que comença a mostrar l'activitat d'`eth0`.

Finalment, si està tot bé haurem de configurar el `cron` perquè s'actualitzin les gràfiques cada 5', per a això editem el `crontab` -e inserint `* /5 * * * *`
`root env LANG=C /usr/bin/mrtg /etc/mrtg.cfg.`

MRTG és molt potent per a visualitzar gràfics de variables ja sigui SNMP o de *scripts* que podem incloure. Vegem dos exemples on el primer l'utilitzarem per a mesurar la càrrega de CPU, basat en una consulta SNMP al servidor, i en el segon executarem un *script* per a veure els processos i els de *root* del sistema (amb diferents opcions de colors, per exemple).


```

Per a la càrrega de CPU, agregar al final de /etc/mrtg:
LoadMIBs: /usr/share/mibs/net/netsnmp/UCD-SNMP-MIB
Target[localhost.cpu]:ssCpuRawUser.0&ssCpuRawUser.0:public@127.0.0.1+
    ssCpuRawSystem.0&ssCpuRawSystem.0:public@127.0.0.1+
ssCpuRawNice.0&ssCpuRawNice.0:public@127.0.0.1
RouterUptime[localhost.cpu]: public@127.0.0.1
MaxBytes[localhost.cpu]: 100
Title[localhost.cpu]: CPU Load
PageTop[localhost.cpu]: <H1>CPU Load %</H1>
Unscaled[localhost.cpu]: ymwd
ShortLegend[localhost.cpu]: %
YLegend[localhost.cpu]: CPU Utilization
Legend1[localhost.cpu]: Active CPU in % (Load)
Legend2[localhost.cpu]:
Legend3[localhost.cpu]:
Legend4[localhost.cpu]:
LegendI[localhost.cpu]: Active
LegendO[localhost.cpu]:
Options[localhost.cpu]: growright,nopercent

```

```

Per al nombre de processos, agregar al final de /etc/mrtg:
Title[procesos]: Processes
Target[procesos]:`/usr/local/bin/proc.sh`
PageTop[procesos]: <h1>Processes</h1>
MaxBytes[procesos]: 200
YLegend[procesos]: Processes
ShortLegend[procesos]: Num.
XSize[procesos]: 300
YSize[procesos]: 100
Options[procesos]: nopercent,gauge
Colours[procesos]: ORANGE\#FF7500,BLUE\#1000ff,DARK GREEN\#006600,VIOLET\#ff00ff
LegendI[procesos]: Processes Root
LegendO[procesos]: Total of Processes

```

On l'script proc.sh és:

```

#!/bin/bash
sysname=`hostname`
p1=`ps -edaf | wc -l`
p1=`expr $p1 - 1`
p2=`ps -edaf | grep ^root | wc -l`
p2=`expr $p2 - 2`

echo $p2
echo $p1
echo $sysname

```

S'ha de tenir en compte d'executar el

```

indexmaker -title=L·Localhost--output /var/www/mrtg/index.html
/etc/mrtg.cfg

```

i després per a verificar que tot està bé

```

env LANG=C /usr/bin/mrtg /etc/mrtg.cfg,

```

recarregant l'URL <http://sysdw.nteum.org/mrtg/> veurem les dues noves gràfiques i la seva activitat.

En les següents referències [34, 10, 11, 17] es pot trobar més informació sobre SNMP (configuració, solució d'errors, etc).

1.2.4. Nagios

Nagios és un sistema de monitoratge de xarxes i sistemes àmpliament utilitzat per la seva versatilitat i flexibilitat, sobretot a l'hora de comunicar alertes de comportaments o tendències dels sistemes monitorats. Pot monitorar serveis de xarxa (SMTP, HTTP, SNMP, etc.) i recursos del sistema (càrrega del processador, ús dels discos, memòria, estat dels ports, etc.) tant locals com remots a través d'un *plugin* (anomenat NRPE) i es pot estendre la seva funcionalitat mitjançant aquests *plugins*. El producte base de Nagios és anomenat **Nagios Core** el qual és *Open Source*, i serveix de base a altres productes comercials de Nagios com NagiosXI, IM i NA. Nagios té una comunitat molt activa (anomenada Nagios Exchange) i a la pàgina web de Nagios (<http://www.nagios.org/>) es pot accedir a les contribucions, *plugins* i documentació. Nagios permet consultar pràcticament qualsevol paràmetre d'interès d'un sistema, i genera alertes, que poden ser rebudes pels responsables corresponents mitjançant diferents canals com per exemple correu electrònic i missatges SMS. La instal·lació bàsica és molt simple, fent `apt-get install nagios3`, que instal·larà totes les biblioteques i *plugins* per a una primera configuració. Durant la instal·lació se'ns sol·licitarà un *passwd*, però també després es pot canviar fent `cd /etc/nagios3; htpasswd htpasswd.users nagiosadmin`. Després podrem recarregar novament Apache2 i connectant a l'URL <http://sysdw.nteum.org/nagios3/> se sol·licitarà l'accés com a *nagiosadmin* i el *passwd* que hem introduït i podrem veure l'estructura de Nagios i observar els serveis monitorats en cadascun dels apartats.

L'arxiu de configuració de Nagios està en `/etc/nagios3/nagios.cfg`, el qual inclou tots els arxius del directori `conf.d` del mateix directori. En aquests tindrem agrupats per arxius els diferents aspectes que cal monitorar, per exemple sistemes (`localhost_nagios2.cfg`) i serveis (`services_nagios2.cfg`). El fitxer més important probablement és `localhost_nagios2.cfg`, del qual farem una breu descripció:

```
# definició d'un host utilitzant un template (generic-host)
define host{
    use                generic-host
    host_name          localhost
    alias              localhost
    address            127.0.0.1
}

# definició d'un servei -espai de disc- utilitzant un template
# (generic-service) executant un cmd i amb advertiment al 20% i error al 10%
# de l'espai lliure.
define service{
    use                generic-service
    host_name          localhost
    service_description Disk Space
    check_command      check_all_disks!20%!10%
}
```

```
# ídem per a usuaris: advertiment 20 usuaris, crític 50 usuaris
define service{
    use                generic-service
    host_name          localhost
    service_description Current Users
    check_command       check_users!20!50
}
#ídem processos:advertència 250, crític 400
define service{
    use                generic-service
    host_name          localhost
    service_description Total Processes
    check_command      check_procs!250!400
}
# Càrrega de CPU.
define service{
    use                generic-service
    host_name          localhost
    service_description Current Load
    check_command      check_load!5.0!4.0!3.0!10.0!6.0!4.0
}
```

Si volguéssim agregar un servei (per exemple ping al *localhost*), només l'hauríem d'agregar al final de l'arxiu:

```
# Ping: advertiment quan el 20% sigui 100 ms, crític 60% sigui 500 ms.
define service{
    use                generic-service
    host              localhost
    service_description PING
    check_command      check_ping!100.0,20%!500.0,60%
}
```

Els *plugins* són incorporats per */etc/nagios3/nagios.cfg* i estan definits en l'arxiu */etc/nagios-plugins/config*, on es poden veure les diferents alternatives per a monitorar.

Dos complements interessants per a Nagios són PNP4Nagios* i NagVis:

*<http://docs.pnp4nagios.org/pnp-0.6/start>

1) PNP4Nagios permet emmagatzemar les informacions que proveeixen els *plugins* en una base de dades anomenada *RRD-database* i cridar l'eina *RRD Tool* (<http://oss.oetiker.ch/rrdtool/>), que permet la visualització d'aquestes dades incrustades a la pàgina de Nagios. Per la qual cosa, a més del valor instantani de les variables que cal monitorar, també tindrem integrats aquests en el temps i podrem veure la seva evolució.

2) NavVis (<http://www.nagvis.org/>) permet dibuixar (secció map) la xarxa de diferents formes i aspectes per a tenir diverses visualitzacions de la xarxa monitorada en forma activa, és a dir, veient els valors de les variables seleccionades sobre el gràfic.

Finalment, és interessant considerar **Icinga** (<https://www.icinga.org/>), que és una bifurcació (*fork*) de Nagios (del 2009) i ha evolucionat molt últimament (per a alguns ja ha superat Nagios) i el seu objectiu és transformar-se en una

eina de referència dins de l'*Open Source* i en l'àmbit del monitoratge de xarxes i sistemes. Icinga neix amb la idea de superar les deficiències en el procés de desenvolupament de Nagios i les seves polítiques, així com la voluntat de ser més dinàmica i fàcil per a agregar-hi noves característiques, com per exemple una interfície d'usuari d'estil Web 2.0, connectors de base de dades addicionals i una API REST que permeti als administradors integrar nombroses extensions sense complicades modificacions del nucli. Icinga està disponible en Debian i la seva configuració és pràcticament similar a Nagios.

1.2.5. Ganglia

Ganglia [12] és una eina que permet monitorar de manera escalable i distribuïda l'estat d'un conjunt de màquines agrupades sota diferents criteris (xarxa, serveis, etc.) o simplement sota una mateixa identificació que anomenarem *clúster*. L'aplicació mostra a l'usuari les estadístiques de forma remota (per exemple, les mitjanes de càrrega de la CPU o la utilització de la xarxa) de totes les màquines que conformen aquest clúster basant-se en un disseny jeràrquic i utilitza comunicacions punt a punt o *multicast* per a l'intercanvi d'informació entre els diferents nodes que formen el clúster. Ganglia utilitza XML per a la representació de dades, XDR per al transport compacte i portàtil de dades i RRDtool (<http://oss.oetiker.ch/rrdtool/>) per a emmagatzematge de dades i visualització. El sistema es compon de dos *daemons* (*gmond* i *gmetad*), una pàgina de visualització (*ganglia-webfrontend*) basada en PHP.

Gmond és un *daemon* multifil que s'executa en cada node del clúster que es desitja supervisar (no és necessari tenir un sistema d'arxius NFS o base de dades ni mantenir comptes especials dels arxius de configuració). Les tasques de Gmond són monitorar els canvis d'estat en el *host*, enviar els canvis pertinents, escoltar l'estat d'altres nodes (a través d'un canal *unicast* o *multicast*) i respondre les peticions d'un XML de l'estat del clúster. La federació dels nodes es porta a terme amb un arbre de connexions punt a punt entre els nodes determinats (representatius) del clúster per a agregar l'estat dels restants nodes. En cada node de l'arbre, s'executa el *daemon* **Gmetad** que periòdicament sol·licita les dades dels restants nodes, analitza el XML, guarda tots els paràmetres numèrics i exporta el XML agregat per un *socket* TCP. Les fonts de dades poden ser *daemons* *gmond*, en representació de determinats grups, o altres *daemons* *gmetad*, en representació de conjunts de grups. Finalment, la web de Ganglia proporciona una vista de la informació recollida dels nodes del clúster en temps real. Per exemple, es pot veure la utilització de la CPU durant l'última hora, dia, setmana, mes o any i mostra gràfics similars per a l'ús de memòria, ús de disc, estadístiques de la xarxa, nombre de processos en execució i tots els altres indicadors de Ganglia.

Per a la instal·lació de Ganglia sobre Debian (és similar per a altres distribucions):

1) Cal fer la instal·lació dels paquets de Ganglia sobre el servidor web: `apt-get install ganglia-monitor gmetad ganglia-webfrontend`.

2) Sobre tots els altres nodes, només es necessita tenir instal·lat el paquet `ganglia-monitor`.

3) Els arxius de configuració estan en `/etc/ganglia` i en `gmond.conf`, la línia més important és `data_source "my cluster" localhost`, on indica quin serà el nom del clúster (per a agregar totes les màquines sota el mateix *tag*) i on es recolliran les dades (en aquest cas, en *localhost*). En `gmond.conf` tenim les configuracions generals i de noms a més dels canals de comunicació que per defecte són *multicast*. Si desitgem que siguin *unicast* haurem de modificar les seccions `udp_send|recv` on la IP de host serà la del servidor (`gmetad`) i comentar les adreces de *multicast*:

```
udp_send_channel {  
# mcast_join = 239.2.11.71  
host = IP_node_gmetad  
port = 8649  
ttl = 1  
}  
  
udp_recv_channel {  
# mcast_join = 239.2.11.71  
port = 8649  
# bind = 239.2.11.71  
}
```

4) Finalment, hem de crear l'enllaç entre la configuració de Ganglia-frontend i Apache fent

```
ln -s /etc/ganglia-webfrontend/apache.conf /etc/apache2/conf.d/ganglia,
```

reiniciem Apache (`service apache2 restart`) i després ens connectem a l'URL <http://sysdw.nteum.org/ganglia/> per a visualitzar una panoràmica global dels sistemes monitorats i fent clic en cada imatge/botó podrem obtenir més informació sobre cadascun dels recursos monitorats.

1.2.6. Altres eines

Altres paquets interessants que cal tenir en compte per a monitorar un sistema són els següents:

- **Zabbix** [35] és un sistema de monitoratge (*Open Source*) que permet recollir l'estat de diferents serveis de xarxa, servidors i maquinari de xarxa. Aquest programa utilitza una base de dades (MySQL, PostgreSQL, SQLite, etc.) per a millorar les prestacions i permet la instal·lació d'agents Zabbix sobre diferents màquines per a monitorar aspectes interns com per exemple càrrega de CPU, utilització de xarxa, espai en disc, etc. També és possible fer aquest monitoratge mitjançant diferents protocols com SNMP, TCP i ICMP, IPMI, etc. i suporta una varietat de mecanismes de notificació en temps real, incloent XMPP. La seva instal·lació no està disponible en els paquets de Debian (per diferències en les polítiques de Debian i Zabbix des del 2012)

però es pot obtenir el paquet (.deb) des del web de Zabbix i seguir la guia d'instal·lació*.

*https://www.zabbix.com/documentation/2.0/manual/installation/install_from_packages

- **Cacti** [16] és una solució gràfica dissenyada per a treballar conjuntament amb dades de RRDtool. Cacti proveeix de diferents formes de gràfiques, mètodes d'adquisició i característiques que pot controlar l'usuari molt fàcilment i és una solució que s'adapta des d'una màquina a un entorn complex de màquines, xarxes i servidors.
- **Frysk** [20], on l'objectiu del projecte és crear un sistema de monitoratge distribuït i intel·ligent per a monitorar processos i fils.

Vegeu també

Cacti es descriu en el mòdul "Clúster, Cloud i DevOps".

Hi ha un conjunt addicional d'eines no menys interessants (no s'inclouen les ja esmentades) que incorpora GNU/Linux per al monitoratge de diferents aspectes del sistema (es recomana veure el `man` de cada eina per a més informació):

- **isag**: *Interactive System Activity Grapher*, per a l'auditoria de recursos hw/sw.
- **mon**: monitor de serveis de xarxa.
- **diffmon**, **fcheck**: generació d'informes sobre canvis en la configuració del sistema i monitoratge dels sistemes de fitxers per a detectar intrusions.
- **fam**: *file alteration monitor*, monitor d'alteració de fitxers.
- **genpower**: monitor per a gestionar les fallades d'alimentació.
- **ksensors (lm-sensors)**: monitor de la placa base (temperatura, alimentació, ventiladors, etc.).
- **systune**: utilitat per a retirar capacitats assignades al nucli en el fitxer `/proc/sys/kernel`.
- **swatch**: monitor per a l'activitat del sistema mitjançant arxius de registre.
- **vtgrab**: monitoratge de màquines remotes (similar a VNC).
- **whowatch**: eina en temps real per al monitoratge d'usuaris.
- **wmnd**: monitor de trànsit de xarxa i monitoratge d'un clúster per xarxa.

1.3. Alta disponibilitat en Linux (*High-Availability Linux*)

Actualment Linux és conegut com un sistema operatiu estable; els problemes es generen quan el maquinari falla. En els casos en què una fallada de maquinari provoca greus conseqüències, a causa de la naturalesa del servei (aplicacions crítiques), s'implementen sistemes tolerants a fallades (*fault tolerant*, FT) amb els quals es garanteix, amb una determinada probabilitat (molt alta), que el servei estigui sempre actiu. El problema d'aquests sistemes és que són extremadament cars, solen ser solucions tancades, totalment dependents de la solució integrada. Els sistemes d'alta disponibilitat (*high availability*, HA) intenten obtenir prestacions properes a la tolerància a fallades, però a costos accessibles. L'alta disponibilitat està basada en la replicació d'elements, per la qual cosa deixarem de tenir un servidor i necessitarem tenir un clúster d'alta disponibilitat. Existeixen per a Linux diferents solucions, com per exemple

Heartbeat (element principal del Linux-HA), ldirectord i LVS (Linux Virtual Server), Piranha (solució basada en LVS de Red Hat), UltraMonkey (solució de VA Linux), o OpenAIS+Corosync+Pacemaker.

El projecte Linux-HA (Linux d'alta disponibilitat) [18] és una solució clúster d'alta disponibilitat per a Linux i altres sistemes operatius, com FreeBSD, OpenBSD, Solaris i MacOSX i que proporciona fiabilitat, disponibilitat i prescripció discontinua de serveis. El producte principal del projecte és **Heartbeat**, l'objectiu principal del qual és la gestió de clústers amb l'objectiu d'obtenir alta disponibilitat. Les seves característiques més importants són les següents: il·limitat nombre de nodes (útil tant per a petits clústers com per a mides grans), monitoratge de recursos (aquests es poden reiniciar o desplaçar a un altre node en cas de fallada), mecanisme de cerca per a eliminar nodes amb fallades del clúster, gestió de recursos basada en directives o regles amb possibilitat d'incloure el temps, gestió preconfigurada de recursos (Apache, DB2, Oracle, PostgreSQL, etc.) i interfície gràfica de configuració. Per a poder ser útils als usuaris, el *daemon* de Heartbeat ha de combinar-se amb un administrador de recursos de clúster (CRM), que té la tasca d'iniciar i aturar els serveis (adreces IP, servidors web, etc.) la qual cosa proporcionarà l'alta disponibilitat. Des de la versió 2.1.3 de Heartbeat s'ha substituït el codi del gestor de recursos del clúster (CRM) pel component Pacemaker. Pacemaker aconsegueix la màxima disponibilitat dels seus serveis de clúster mitjançant la detecció i recuperació de nodes i les fallades de nivell de servei. Això s'aconsegueix mitjançant la utilització de les capacitats de missatgeria i la pertinença a la infraestructura proporcionada OpenAIS|Corosync + Pacemaker [25].

1.3.1. Guia breu d'instal·lació de Heartbeat i Pacemaker (Debian)

En aquest subapartat es donarà una breu descripció de com construir un clúster d'alta disponibilitat de dos nodes amb Heartbeat* per a, per exemple, disposar d'un servidor Apache d'alta disponibilitat. És interessant (encara que una mica antic) l'article [30] i la documentació del lloc web de Linux-HA [19]. Com a punt inicial, disposem de dos ordinadors similars (no és necessari, però si un ha d'ocupar el lloc de l'altre és aconsellable). Els servidors els anomenarem *NteumA* (primari) i *VteumB* (secundari), amb una interfície a xarxa que utilitzaran per a connectar-se a la xarxa tant entre ells com des de fora. Per a fer aquesta prova de concepte, hem utilitzat dues màquines virtuals (amb la xarxa configurada en mode *bridged* perquè es vegin a la xarxa), però és la mateixa prova amb dues màquines físiques. Les nostres màquines estan configurades com *NteumA*: `eth0 = 192.168.1.201` i *VteumB*: `eth0 = 192.168.1.202`, les dues tenen com a *netmask* `255.255.255.0` i *gateway* `192.168.1.1`. I definim, a més, una adreça virtual on prestarem els serveis, per exemple Apache `192.168.1.200`. Configurem les màquines perquè es vegin entre ells (a través d'un `ping`) i només és necessari que tinguin una instal·lació mínima amb `apache` i `sshd` i que estiguin configurades tant en nom (`hostname`) com en `/etc/hosts` amb IP `FDQN` àlies (per exemple, que per a cada màquina hi ha-

*La informació detallada es pot consultar a

`file:///usr/share
/doc/heartbeat
/GettingStarted.html.`

gi una línia com 192.168.1.201 nteuma.nteu.org nteuma per a totes les màquines del clúster i haurem de veure el nom de la màquina com l'hem configurat en */etc/hosts* en el nom curt, per exemple amb `uname -n`). Per a instal·lar i configurar Apache2 + Heartbeat fem:

- 1) `apt-get install apache2`
- 2) Modifiquem la configuració d'Apache agregant a *etc/apache2/ports.conf* la línia `NameVirtualHost 192.168.1.200:80` (deixant les restants com estan).
- 3) Perquè Apache no estigui arrencat des de l'inici (ja que volem que el faci Heartbeat) el traiem dels *scripts rc*.d* amb `update-rc.d apache2 remove`.
- 4) Instal·lem el paquet `chkconfig` que després necessitarà Heartbeat `apt-get install chkconfig`.
- 5) Instal·lem el paquet Heartbeat amb `apt-get install heartbeat`. Totes aquestes accions anteriors les fem a les dues màquines.
- 6) Sobre nteuma (primari), fem la configuració de Heartbeat. Els fitxers de configuració estan en */etc/ha.d/* i les plantilles per a configurar-los estan en */usr/share/doc/heartbeat/*. Per a l'arxiu *ha.cf*, el modifiquem amb el següent:

```
logfile /var/log/cluster.log
logfacility local0
warntime 5
deadtime 30
initdead 120
keepalive 2
bcast eth0
udpport 694
auto_failback on
node nteuma
node vteumb
```

On *logfile* i *logfacility* indiquen on estarà l'arxiu de *log* i el nivell de missatges que volem, *warntime* és el temps que transcorrerà abans que Heartbeat ens avisi, *deadtime* el temps després del qual Heartbeat confirmarà que un node ha caigut, *initdead* el temps màxim que Heartbeat esperarà al fet que un node arrenqui, i *keepalive* l'interval de temps per a comprovar la disponibilitat. A més, *bcast* la forma de comunicació (*broadcast*) i la interfície i *node* els nodes que forma el nostre clúster HA.

- 7) L'arxiu *authkeys* és on es configura la comunicació entre els nodes del clúster que haurà de tenir permisos només de *root*
(`chmod 600 /etc/ha.d/authkeys`):

```
auth 2
2 sha1 MyPaSSWoRd
```

- 8) a l'arxiu */etc/ha.d/haresources* li indiquem el node primari i el servei (apache2) que cal aixecar:

```
nteuma IPaddr2::192.168.2.100/24/eth0 apache2
```


9) Ara hem de propagar la configuració als nodes del clúster (vteumb en el nostre cas, però es faria a tots els nodes indicats en ha.cf):

```
/usr/share/heartbeat/ha_propagate.
```

10) Reiniciem les màquines (*reboot*) per a assegurar-nos que tot s'inicia com desitgem.

Per a comprovar com és el servidor que està prestant el servei, hem inclòs en */var/www/index.html* el nom de la màquina. Així, quan carreguem la pàgina a través de la connexió a la IP 192.168.1.200 veurem quina màquina és la que presta el servei. En primer lloc, haurem de veure què és NteumA i si traiem la xarxa d'aquesta (*ifdown eth0*) veurem que en uns segons en actualitzar la pàgina serà VteumB.

Cal consultar la documentació en l'adreça <http://www.linux-ha.org/doc/man-pages/man-pages.html>; i com a ordres per a consultar i veure l'estat del nostre clúster podem utilitzar el següent: *cl_status* per a veure l'estat de tot el sistema, *hb_addnode* per a enviar un missatge al clúster per a afegir nous nodes o *hb_delnode*, per a treure'ls.

Es pot agregar en *ha.cf* l'ordre *crm respawn* (que és el *cluster resource manager* -crm- de LinuxHA), que iniciarà el *daemon* *crmd* que ens brindarà informació sobre l'estat del clúster i ens permetrà gestionar amb una sèrie d'ordres (*crm_**) els recursos del clúster. Per exemple, *crm_mon -l* ens donarà informació de l'estat del nostre clúster.

Com podrem comprovar si agreguem aquesta línia, el nostre simple clúster deixarà de funcionar, ja que el control de recurs ara l'està fent Pacemaker (*crm*) i haurem de configurar-lo amb aquesta finalitat (o treure la línia *crm respawn* i reiniciar els serveis). Per a més informació, podeu consultar [21, 22, 23, 24].

Per a configurar el CRM juntament amb Heartbeat per a un servei actiu-passiu on si un node falla el node passiu adoptarà la seva funció (en el nostre cas, Apache). En el nostre cas, utilitzarem Heartbeat per a mantenir la comunicació entre els nodes i Pacemaker com a *resource manager* que proporciona el control i administració dels recursos proveïts pel clúster. Bàsicament, Pacemaker pot manejar diferents tipus de recursos anomenats LSB, que seran els que proporciona GNU/Linux i que es poden gestionar mitjançant */etc/init.d* i OCF, que ens permetrà inicialitzar una adreça virtual, monitorar un recurs, iniciar/parar un recurs, canviar el seu ordre de servei, etc.

1) Partim del fet que ja hem instal·lat Heartbeat i Pacemaker (aquest últim s'instal·la quan s'instal·la Heartbeat) i, si no, podem fer

```
apt-get install heartbeat pacemaker
```

2) A la configuració de `/etc/ha.d/ha.cf` esmentada anteriorment agreguem `crm respawn` per a iniciar Pacemaker (veurem que hi ha un procés anomenat `crmd`).

3) `/etc/ha.d/authkeys` el deixem com ja el teníem configurat.

4) Reiniciem el servei `service heartbeat restart`.

5) Ara podrem veure l'estat del clúster amb `crm status`.

```
=====
Last updated: Tue Jul 1 12:06:36 2014
Stack: Heartbeat
Current DC: vteumb (...) - partition with quorum
...
2 Nodes configured, unknown expected votes
0 Resources configured.
=====
Online: [ veteumb nteuma ]
```

6) Deshabilitem stonith ja que no el necessitarem: `crm configure property stonith-enabled=false`

7) Inicialitzem els nodes per al quòrum:

```
crm configure property expected-quorum-votes="2"
```

8) Per a tenir quòrum, la meitat dels nodes del clúster han de ser en línia (nombre de nodes/2)+1, però en un clúster de 2 nodes això no ocorre quan falla un node, i per tant necessitem inicialitzar la política a *ignore*: `crm configure property no-quorum-policy=ignore`

9) Per a preveure el *failback* d'un recurs: `crm configure rsc_defaults resource-stickiness=100`

10) Podem fer una llista dels objectes OCF amb `crm ra list ocf` i en el nostre cas utilitzarem `IPaddr2`. Per a obtenir més informació sobre aquest: `crm ra info ocf:IPaddr2`

11) Agreguem una IP virtual (VIP) al nostre clúster:

```
crm configure primitive havip1 ocf:IPaddr2 params ip=192.168.1.200
cidr_netmask=32 nic=eth0 op monitor interval=30s
```

12) Verifiquem que el recurs `havip1` es troba en el primer node executant `crm status` i ens donarà una informació similar a l'anterior però amb una línia com *havip1 (ocf::heartbeat:IPaddr2): Started nteuma*

13) Agreguem el *daemon* al nostre clúster: `crm ra info ocf:anything`

14) Agreguem Apache al nostre clúster

```
crm configure primitive apacheha lsb::apache2 op monitor interval=15s
```

15) Inicialitzem el recurs VIP i Apache en el mateix node: `crm configure colocation apacheha-havip1 INFINITY: havip1 apacheha` i veurem amb `crm status` que *havip1 (ocf::heartbeat:IPaddr2): Started nteuma apacheha (lsb:apache2): Started nteuma*

16) Podem configurar l'ordre dels serveis:

```
crm configure order ip-apache mandatory: havip1 apacheha
```

17) O migrar un servei a un altre node: `crm resource migrate apacheha vteumb` (amb la qual cosa, si ens connectem a 192.168.1.200 veurem que el servei és proporcionat per un servidor o per un altre). Aquesta ordre la podem utilitzar per a fer manteniments transferint el servei d'un lloc a un altre sense interrompre'l. Si executem `crm status` veurem

```
havip1 (ocf::heartbeat:IPaddr2): Started nteuma apacheha (lsb:apache2): Started vteumb
```

18) La configuració del nostre clúster la podem veure amb `crm configure show`:

19) És interessant executar el navegador amb l'URL 192.168.1.200 i anar desactivant els nodes o recuperant-los per a veure com adopta el rol cadascun i mantenen el servei (mireu l'estat entre canvi i canvi amb `crm status`).

Si comentem la línia `crm respawn` en `/etc/ha.d/ha.cf`, tornarem a la gestió bàsica anterior sense el gestor de recursos Pacemaker.

1.3.2. DRBD

El **Distributed Replicated Block Device (DRBD)** és un emmagatzematge distribuït sobre múltiples *hosts* on, com en RAID1, les dades són replicades sobre el sistema d'arxiu sobre els altres *hosts* i sobre TCP/IP.[26] En aquesta prova de concepte, utilitzarem les mateixes màquines fetes servir en Heartbeat a les quals hem addicionat una unitat de disc més (`/dev/sdb`) i hem creat una partició sobre aquestes (`/dev/sdb1`). La instal·lació és:

1) Instal·lar a les dues màquines DRBD `apt-get install drbd8-utils` on els arxius de configuració estaran en `/etc/drbd.d/` (hi ha un arxiu `/etc/drbd.conf` però que inclou els arxius del directori esmentat).

2) Crearem la configuració del recurs com `/etc/drbd.d/demo.res` amb el contingut següent:

```
resource drbddemo {
    meta-disk internal;
    device /dev/drbd1;
    syncer {
        verify-alg sha1;
    }
    net {
        allow-two-primaries;
    }
    on nteuma {
        disk /dev/sdb1;
        address 192.168.1.201:7789;
    }
    on vteumb {
```

```

        disk /dev/sdb1;
        address 192.168.1.202:7789;
    }
}

```

3) Copiem l'arxiu:

```
scp /etc/drbd.d/demo.res vteum:/etc/drbd.d/demo.res
```

4) Inicialitzem en els dos nodes el dispositiu executant `drbdadm create-md drbddemo`

5) Sobre `vteum` (s'ha de verificar amb `uname -n`) fem `modprobe drbd` per a carregar el mòdul de kernel, després `drbdadm up drbddemo` per a aixecar el dispositiu i el podrem mirar amb `cat /proc/drbd` que no indicarà (entre altres missatges) que està:

```
cs:WfConnection ro:Secondary/Unknown ds:Inconsistent/DUnknown C r—
```

6) Sobre `vteumb` fem `modprobe drbd` per a carregar el mòdul del kernel, `drbdadm up drbddemo` per a inicialitzar el dispositiu i

```
drbdadm --overwrite-data-of-peer primary drbddemo
```

per a configurar-lo com a primari. Si mirem la configuració amb l'ordre `cat /proc/drbd`, veurem entre altres missatges

```
1: cs:SyncSource ro:Primary/Secondary ds:UpToDate/Inconsistent C r— i més a sota
[>.....] sync'ed: 0.4
```

7) A les dues màquines veurem el dispositiu `/dev/drbd1`, però només des d'aquell que és primari el podrem muntar (no des del secundari). Per a això, instal·lem les eines per a crear un *XFS filesystem* amb l'ordre `apt-get install xfsprogs` i fem sobre el primari (`vteumb`) `mkfs.xfs /dev/drbd1` i després el podrem muntar amb `mount /dev/drbd1 /mnt` i podrem copiar un arxiu com `cp /var/log/messages /mnt/test`.

8) Per a executar uns petits tests, podem fer sobre `vteum`: `umount /mnt` i després `drbdadm secondary drbddemo` i sobre `vteum` primer hem d'instal·lar XFS `apt-get install xfsprogs`, després `drbdadm primary drbddemo` i finalment `mount -t xfs /dev/drbd1 /mnt`. Podem comprovar el contingut amb `ls /mnt` i veurem els arxius que copiem en `vteum`. Amb l'ordre `cat /proc/drbd` a ambdues màquines veurem l'intercanvi de rols primari per secundari i viceversa.

9) Un segon test que podem fer és apagar el node `vteum` (secundari) per a simular una fallada i veurem amb `cat /proc/drbd` que l'altre node no està disponible (0: `cs:WfConnection ro:Primary/Unknown`). Si copiem, fem `cp /mnt/test /mnt/test1` i posem en marxa `vteum`, quan faci el *boot* sincronitzarà les dades i veurem sobre `vteum` 0: `cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r—` i sobre `vteum` estaran els arxius sincronitzats.

10) Si provoquem una fallada sobre el primari (apagant-lo o desconnectant la xarxa) i tenim muntat el directori (és el que coneixem com a *shutdown* no ordenat) no hi haurà problemes per a anar a l'altre node, canviar-lo a primari

i muntar-lo. Si recuperem el node primari que va fallar, veurem que tots dos queden com a secundaris quan torna a estar actiu, per la qual cosa l'haurem de posar com a primari i el podrem muntar novament, i en el secundari veurem que s'ha sincronitzat i després en uns segons tornarà al seu rol de secundari/primari.

11) Si hi ha canvis durant el temps que un node ha estat desconnectat com a primari, obtindrem un missatge de “split-brain” i per a recuperar-lo haurem d'executar sobre el secundari

```
drbdadm secondary drbddemo i drbdadm --discard-my-data connect drbddemo
```

i sobre el primari `drbdadm connect drbddemo` per a connectar els nodes i resoldre el conflicte. [27]

1.3.3. DRBD + Heartbeat com a NFS d'alta disponibilitat

L'objectiu d'aquest apartat és mostrar com utilitzar DRBD i Heartbeat per a generar un clúster NFS que permeti tenir una còpia del NFS i que entri en servei quan el servidor primari deixi de funcionar. Per a això, utilitzarem la mateixa instal·lació de l'apartat anterior però sobre un altre disc en cada node (sdc) sobre el qual hem creat una partició (*/dev/sdc1*), i és important verificar que tenim els mòduls de DRBD instal·lats (`lsmod | grep drbd`) [28, 29]. A continuació, seguim aquests passos:

1) Crearem un arxiu */etc/drbd.d/demo2.res* amb el següent contingut:

```
resource myrs {
    protocol C;
    startup { wfc-timeout 0; degr-wfc-timeout 120; }
    disk { on-io-error detach; }
    on nteuma {
        device /dev/drbd2;
        disk /dev/sdc1;
        meta-disk internal;
        address 192.168.1.201:7788;
    }
    on vteumb {
        device /dev/drbd2;
        disk /dev/sdc1;
        meta-disk internal;
        address 192.168.1.202:7788;
    }
}
```

On utilitzem la partició de cada disc sobre cada node i generarem el dispositiu */etc/drbd2*. És important que els noms de les màquines siguin exactament els que ens dona `uname -n` i modificar */etc/hosts*, */etc/resolv.conf* i */etc/hostname* perquè les màquines tinguin connectivitat entre elles (i amb l'exterior) mitjançant el nom i la IP. Això s'ha de portar a terme sobre les dues màquines, incloent-hi la còpia de l'arxiu anterior.

2) També sobre les dues màquines haurem d'executar `drbdadm create-md myrs` per a inicialitzar el dispositiu, `drbdadm up myrs` per a activar-lo i l'ordre

`drbdadm syncer myrs` per a sincronitzar-lo. Podrem veure el resultat amb `cat /proc/drbd`, que ens mostrarà els dispositius com a "Connected" i inicialitzats.

3) Sobre el servidor primari executem `drbdadm --overwrite-data-of-peer primary myrs` per a indicar-li que sigui primari, i visualitzant l'arxiu `/proc/drbd` veurem el resultat.

4) Finalment, executem/reiniciem el servei `service drbd start|restart` amb la qual cosa tindrem un dispositiu `/dev/drbd2` preparat per a configurar el sistema d'arxiu.

5) En aquest cas, utilitzarem LVM, ja que permet més flexibilitat per a gestionar les particions, però es podria utilitzar `/dev/drbd2` com a dispositiu de blocs simplement. Instal·lem LVM (`apt-get install lvm2`) i executem:

<code>pvcreate /dev/drbd2</code>	Creem la partició LVM física
<code>pvdisplay</code>	Visualitzem
<code>vgcreate myrs /dev/drbd2</code>	Creem el grup anomenat myrs
<code>lvcreate -L 20 M -n web_files myrs</code>	Creem una partició lògica web_files
<code>lvcreate -L 20 M -n data_files myrs</code>	Creem una altra partició lògica data_files
<code>lvdisplay</code>	Visualitzem

Amb això, haurem de tenir disponibles les particions en `/dev/myrs/web_files` i `/dev/myrs/data_files`.

Amb això, crearem el sistema d'arxius i el muntem:

<code>mkfs.ext4 /dev/myrs/web_files</code>	
<code>mkfs.ext4 /dev/myrs/data_files</code>	
<code>mkdir /data/web-files</code>	Creem els punts de muntatge
<code>mkdir /data/data-files</code>	
<code>mount /dev/myrs/web_files /data/web-files</code>	Muntem les particions
<code>mount /dev/myrs/data_files /data/data-files</code>	

6) Ara haurem d'instal·lar i configurar el servidor NFS (sobre els dos servidors) perquè pugui ser gestionat per Heartbeat i exportar-lo als clients. Per a això, executem (`apt-get install nfs-kernel-server`) i editem l'arxiu `/etc/exports` amb el següent contingut:

```
/data/web-files 192.168.1.0/24(rw,async,no_root_squash,no_subtree_check,fsid=1)
/data/data-files 192.168.1.0/24(rw,async,no_root_squash,no_subtree_check,fsid=2)
```

És important el valor del paràmetre `fsid` ja que amb aquest els clients de sistema d'arxiu sobre el servidor primari sabran que són els mateixos que en el servidor secundari, i si el primari queda fora no es bloquejaren esperant que torni a estar actiu sinó que continuaran treballant amb el secundari. Com que deixarem que Heartbeat gestioni el NFS, l'hem de treure de *boot* amb

```
update-rc.d -f nfs-common remove i update-rc.d -f nfs-kernel-server
remove.
```

7) Finalment, hem de configurar Heartbeat i per a això modifiquem l'arxiu `/etc/ha.d/ha.cf` amb:

```
autojoin none
auto_failback off
keepalive 2
warntime 5
deadtime 10
```

```
initdead 20
bcast eth0
node nteuma
node vteumb
logfile /var/log/ha-log
debugfile /var/log/had-log
```

S'ha indicat `auto_failback = off`, ja que no desitgem que torni a l'original quan el primari retorni (la qual cosa podria ser desitjable si el hw del primari fos millor que el del secundari, i en aquest cas s'hauria de posar a *on*). `deadtime` indica que considera el servidor fora de servei després de 10 segons, i cada 2 segons preguntarà si estan vius. Deixem el fitxer `/etc/ha.d/authkeys` com ja el teníem definit i executem `/usr/share/heartbeat/ha_propagate` per a copiar els arxius a l'altre servidor (també es podria fer manualment).

8) Per a indicar una adreça virtual als clients NFS, utilitzarem 192.168.1.200 i així sempre tindran aquesta IP com a referent, de manera independent de qui els estigui prestant el servei. El següent és modificar l'arxiu `/etc/ha.d/haresources` per a indicar a Heartbeat quins són els serveis que cal gestionar (IPV, DRBD, LVM2, Filesystems i NFS), i que haurem d'introduir en l'ordre en què es necessiten:

```
nteum \
IPaddr::192.168.1.200/24/eth0 \
drbddisk::myrs \
lvm2 \
Filesystem::/dev/myrs/web_files::/data/web-files::ext4::nosuid,usrquota,noatime \
Filesystem::/dev/myrs/data_files::/data/data-files::ext4::nosuid,usrquota,noatime \
nfs-common \
nfs-kernel-server
```

S'ha de copiar aquest arxiu en els dos servidors (sense modificar), ja que aquest indica que `nteuma` és el servidor primari i quan falli serà `vteumb`, tal com ho hem explicat en `ha.cf`.

9) Finalment, podrem iniciar/reiniciar Heartbeat (amb `service heartbeat start|restart`) i provar a muntar el servidor en una mateixa xarxa i fer les proves de fallada corresponents.

Activitats

1. Feu un monitoratge complet del sistema amb les eines que considereu adequades i efectueu un diagnòstic de la utilització de recursos i colls d'ampolla que podrien existir al sistema. Heu de simular la càrrega al sistema del codi de `sumdis.c` que hem donat en el mòdul "Clúster, Cloud i DevOps". Per exemple, utilitzeu `sumdis 1 2000000`
2. Canvieu els paràmetres del nucli i del compilador i executeu el codi esmentat en l'activitat anterior (`sumdis.c`) amb, per exemple: `time ./sumdis 1 1000000`.
3. Amb l'execució de les dues activitats anteriors, traieu conclusions sobre els resultats.
4. Amb el programa d'iteracions indicat en aquest mòdul, implementeu una manera d'acabar l'execució en 5 segons, independent de les crides al sistema excepte si només és una vegada.
5. Monitoreu tots els programes anteriors amb Munin i Ganglia traient conclusions sobre la seva funcionalitat i prestacions.
6. Registreu quatre serveis amb Monin i feu la gestió i seguiment per mitjà del programa.
7. Instal·leu MRTG i monitoreu la CPU de l'execució dels programes anteriors.
8. Instal·leu i experimenteu amb els sistemes descrits d'alta disponibilitat (Hearbeat, Pacemaker, DRBD).
9. Amb Heartbeat + DRBD, podeu crear un sistema d'arxius NFS redundant i fer les proves de fallada de xarxa sobre el servidor primari (deshabilitant/habilitant la xarxa) perquè el servidor secundari adquireixi el control i després retorni el control a aquest quan el primari recuperi la xarxa.

Bibliografia

- [1] **Eduardo Ciliendo; Takechika Kunimasa** (2007). *Linux Performance and Tuning Guidelines*.
<<http://www.redbooks.ibm.com/redpapers/pdfs/redp4285.pdf>>
- [2] **Nate Wiger**. *Linux Network Tuning for 2013*.
<<http://www.nateware.com/linux-network-tuning-for-2013.html>>
- [3] **Bravo E., D.** (2006). *Mejorando la Performance en Sistemas Linux/Unix*. GbuFDL1.2. <die-gobravoestrada@hotmail.com>
<<http://es.tldp.org/Tutoriales/doc-tut-performance/perf.pdf>>
- [4] **Mourani, G.** (2001). *Securing and Optimizing Linux: The Ultimate Solution*. Open Network Architecture, Inc.
- [5] *Optimització de servidors Linux*.
<http://people.redhat.com/alikins/system_tuning.html>
- [6] *Performance Monitoring Tools for Linux*.
<<http://www.linuxjournal.com/article.php?sid=2396>>
- [7] *Monit*.
<<http://mmonit.com/monit/>>
- [8] *Munin*.
<<http://munin-monitoring.org/>>
- [9] *MRTG*.
<<http://oss.oetiker.ch/mrtg/>>
- [10] **M. Rushing**. *Fix for MRTG Generating SNMP_Session Error in Debian Wheezy*.
<http://mark.orbum.net/2013/06/07/fix-for-mrtg-generating-snmp_session-error-in-debian-wheezy-and-possibly-ubuntu/>
- [11] *SNMP - Debian*.
<<https://wiki.debian.org/SNMP>>
- [12] *Ganglia Monitoring System*.
<<http://ganglia.sourceforge.net/>>
- [13] *Monitorització amb SNMP i MRTG*.
<http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch22_:_Monitoring_Server_Performance>
- [14] **D. Valdez**. *Configuración rápida de MRTG*.
<<http://sysnotas.blogspot.com.es/2013/06/mrtg-configuracion-rapida-para-debian.html>>
- [15] *Howto sobre instalación de MRTG en Debian*.
<<http://preguntaslinux.org/-howto-instalacion-de-mrtg-monitoreo-debian-t-3061.html>>
- [16] *Cacti*.
<<http://cacti.net/>>
- [17] *Net-SNMP - MIBs*.
<<http://www.net-snmp.org/docs/readmefiles.html>>
- [18] *Linux-HA*.
<http://linux-ha.org/wiki/Main_Page>
- [19] *Documentació de Linux-HA*.
<<http://www.linux-ha.org/doc/users-guide/users-guide.html>>
- [20] *Frysk*.
<<http://sources.redhat.com/frysk/>>
- [21] *Pacemaker documentation*.
<<http://clusterlabs.org/doc/>>
- [22] *Pacemaker Cluster From Scratch*.
<http://clusterlabs.org/doc/en-US/Pacemaker/1.0/pdf/Clusters_from_Scratch/Pacemaker-1.0-Clusters_from_Scratch-en-US.pdf>

- [23] **I. Mora Perez.** *Configuring a failover cluster with heartbeat + pacemaker.*
<<http://opentodo.net/2012/04/configuring-a-failover-cluster-with-heartbeat-pacemaker/>>
- [24] **F. Diaz.** *Alta Disponibilidad con Apache2 y Heartbeat en Debian Squeeze.*
<<http://www.muspells.net/blog/2011/04/alta-disponibilidad-con-apache2-y-heartbeat-en-debian-squeeze/>>
- [25] **Pacemaker.**
<http://clusterlabs.org/doc/en-US/Pacemaker/1.0/html/Pacemaker_Explained/s-intro-pacemaker.html>
- [26] **DRBD.**
<<http://www.drbd.org/home/what-is-drbd/>>
- [27] **DRBD tests.**
<<https://wiki.ubuntu.com/Testing/Cases/UbuntuServer-drbd>>
- [28] **R. Bergsma.** *Redundant NFS using DRBD+Heartbeat.*
<<http://blog.remibergsma.com/2012/09/09/building-a-redundant-pair-of-linux-storage-servers-using-drbd-and-heartbeat/>>
- [29] **G. Armer.** *Highly Available NFS Cluster on Debian Wheezy.*
<<http://sigterm.sh/2014/02/highly-available-nfs-cluster-on-debian-wheezy/>>
- [30] **Leung, C. T.** *Building a Two-Node Linux Cluster with Heartbeat.*
<<http://www.linuxjournal.com/article/5862>>
- [31] **Majidimehr, A.** (1996). *Optimizing UNIX for Performance.* Prentice Hall.
- [32] **Monitor Debian servers with monit.**
<<http://www.debian-administration.org/articles/269>>
- [33] **Monitorització amb Munin i monit.**
<http://www.howtoforge.com/server_monitoring_monit_munin>
- [34] **Monitorización con SNMP y MRTG.**
<http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch22_:_Monitoring_Server_Performance>
- [35] **The Enterprise-class Monitoring Solution for Everyone.**
<<http://zabbix.com>>

Clúster, Cloud i DevOps

Remo Suppi Boldrito

PID_00212468

Índex

Introducción	5
Objectius	8
1. Clusterització	9
1.1. Virtualització	9
1.1.1. Plataformes de virtualització	10
1.2. Beowulf	12
1.2.1. Com es configuren els nodes?	13
1.3. Beneficis del còmput distribuït	15
1.3.1. Com cal programar per a aprofitar la concurrència? ..	17
1.4. Memòria compartida. Models de fils (<i>threading</i>)	18
1.4.1. Multifils (<i>multithreading</i>)	19
1.5. OpenMP	22
1.6. MPI, <i>Message Passing Interface</i>	26
1.6.1. Configuració d'un conjunt de màquines per a fer un clúster adaptat a OpenMPI	27
1.7. Rocks Cluster	30
1.7.1. Guia ràpida d'instal·lació	31
1.8. FAI	32
1.8.1. Guia ràpida d'instal·lació	33
1.9. <i>Logs</i>	35
1.9.1. Octopussy	37
1.9.2. Eines de monitoratge addicionals	38
2. Cloud	40
2.1. Opennebula	43
3. DevOps	48
3.1. Linux Containers, LXC	49
3.2. Docker	52
3.3. Puppet	54
3.3.1. Instal·lació	55
3.4. Chef	59
3.5. Vagrant	61
Activitats	64
Bibliografia	65

Introducción

Els avenços en la tecnologia han portat, d'una banda, al desenvolupament de processadors més ràpids, amb més d'un element de còmput (nuclis o *cores*) en cadascun, de baix cost i amb suport a la virtualització maquinari i, d'una altra, al desenvolupament de xarxes altament eficients. Això, juntament amb el desenvolupament de sistemes operatius com GNU/Linux, ha afavorit un canvi radical en la utilització de sistemes de processadors de nuclis múltiples i altament interconnectats, en lloc d'un únic sistema d'alta velocitat (com els sistemes vectorials o els sistemes de processament simètric SMP). A més, aquests sistemes han tingut una corba de desplegament molt ràpida, ja que la relació preu/prestacions ha estat, i ho és cada dia més, molt favorable tant per al responsable dels sistemes TIC d'una organització com per als usuaris finals en diferents aspectes: prestacions, utilitat, fiabilitat, facilitat i eficiència. D'altra banda, les creixents necessitats de còmput (i d'emmagatzematge) s'han convertit en un element tractor d'aquesta tecnologia i el seu desenvolupament, vinculats a la provisió dinàmica de serveis, contractació d'infraestructura i utilització per ús -fins i tot en minuts- (i no per compra o per lloguer), la qual cosa ha generat una evolució total i important en la forma de dissenyar, gestionar i administrar aquests centres de còmput. No menys importants han estat els desenvolupaments, a més dels sistemes operatius, per a suportar totes aquestes tecnologies, en llenguatges de desenvolupament, API i entorns (*frameworks*) perquè els desenvolupadors puguin utilitzar la potencialitat de l'arquitectura subjacent per al desenvolupament de les seves aplicacions, perquè els administradors i gestors puguin desplegar, oferir i gestionar serveis contractats i valorats per minuts d'ús o bytes d'E/S, per exemple, i perquè els usuaris finals puguin fer un ús ràpid i eficient dels sistemes de còmput sense preocupar-se gens d'allò que existeix per sota, on està situat i com s'executa.

Per aquest motiu, en aquest apartat, es desenvolupen tres aspectes bàsics que són part de la mateixa idea, en diferents aspectes, quan es desitja oferir la infraestructura de còmput d'altres prestacions. O una plataforma com a servei o la seva utilització per un usuari final en el desenvolupament d'una aplicació, que permeti utilitzar totes aquestes infraestructures de còmput distribuïdes i d'altres prestacions. En sistemes de còmput d'altres prestacions (*High Performance Computing*-HPC), podem distingir dues grans configuracions:

1) Sistemes fortament acoblats (*tightly coupled systems*): són sistemes on la memòria és compartida per tots els processadors (*shared memory systems*) i la memòria de tots aquests "es veu" (per part del programador) com una única memòria.

2) Sistemes feblement acoblats (*loosely coupled systems*): no comparteixen memòria (cada processador posseeix la seva) i es comuniquen mitjançant missatges passats a través d'una xarxa (*message passing systems*).

En el primer cas, són coneguts com a *sistemes paral·lels de còmput (parallel processing system)* i en el segon, com a *sistemes distribuïts de còmput (distributed computing systems)*.

En l'actualitat, la gran majoria de sistemes (bàsicament per la seva relació preu-prestacions) són del segon tipus i es coneixen com a clústers on els sistemes de còmput estan interconnectats per una xarxa de gran amplada de banda i tots treballen en estreta col·laboració, i l'usuari final els veu com un únic equip. És important indicar que cadascun dels sistemes que integra el clúster al seu torn pot tenir més d'un processador amb més d'un *core* en cadascun, per la qual cosa el programador haurà de tenir en compte aquestes característiques quan desenvolupi les seves aplicacions (llenguatge, memòria compartida/distribuïda, nivells de caché, etc.) i així podrà aprofitar tota la potencialitat de l'arquitectura agregada. El tipus més comú de clúster és l'anomenat Beowulf, que és un clúster implementat amb múltiples sistemes de còmput (generalment similars, però poden ser heterogenis) i interconnectats per una xarxa d'àrea local (generalment Ethernet, però hi ha xarxes més eficients com Infiniband o Myrinet). Alguns autors el denominen *MPP (massively parallel processing)* quan és un clúster, però disposen de xarxes especialitzades d'interconnexió (mentre que els clústers utilitzen maquinari estàndard a les seves xarxes) i estan formats per un alt nombre de recursos de còmput (1.000 processadors, no més). Avui dia, la majoria de sistemes publicats en el TOP500 (<http://www.top500.org/system/177999>) són clústers, i en l'última llista publicada (2014) ocupava el primer lloc l'ordinador Tianhe-2 (Xina) amb 3,1 milions de *cores*, 1 petabyte de memòria RAM (1.000 Tbytes) i un consum de 17 MW.

L'altre concepte vinculat als desenvolupaments tecnològics esmentats a les noves formes d'entendre el món de les TIC en l'actualitat és el *cloud computing* (o informàtica/serveis en el núvol), que sorgeix com a concepte d'oferir serveis d'informàtica a través d'Internet en què l'usuari final no té coneixement d'on s'estan executant les seves aplicacions i tampoc necessita ser un expert per a contactar, pujar i executar les seves aplicacions en minuts. Els proveïdors d'aquest tipus de servei tenen recursos (generalment distribuïts a tot el món) i permeten que els seus usuaris contractin i gestionin aquests recursos en línia i sense la major intervenció i en forma gairebé automàtica, la qual cosa permet reduir els costos tenint avantatges (a més no s'ha d'instal·lar-mantenir la infraestructura física –ni l'obra civil) com la fiabilitat, flexibilitat, rapidesa en l'aprovisionament, facilitat d'ús, pagament per ús, contractació d'allò que es necessita, etc. Òbviament, també té els seus desavantatges, com són la dependència d'un proveïdor i la centralització de les aplicacions/emmagatzematge

de dades, servei vinculat a la disponibilitat d'accés a Internet, qüestions de seguretat, ja que les dades sensibles del negoci no resideixen en les instal·lacions de les empreses i poden generar riscos de sostracció/robatori d'informació, confiabilitat dels serveis prestats pel proveïdor (sempre és necessari signar una SLA –contracte de qualitat de servei), tecnologia susceptible al monopoli, serveis estàndard (només els oferts pel proveïdor), escalabilitat o privadesa.

Un tercer concepte vinculat a aquestes tecnologies, però probablement més del costat dels desenvolupadors d'aplicacions i *frameworks*, és el de DevOps. DevOps sorgeix de la unió de les paraules *Development* (desenvolupament) i *Operations* (operacions), i es refereix a una metodologia de desenvolupament de programari que se centra en la comunicació, col·laboració i integració entre desenvolupadors de programari i els professionals d'operacions / administradors a les tecnologies de la informació (TI). DevOps es presenta com una metodologia que dona resposta a la relació entre el desenvolupament del programari i les operacions TI, tenint com a objectiu que els productes i serveis de programari desenvolupats per una entitat es puguin fer més eficientment, tinguin una alta qualitat i a més siguin segurs i fàcils de mantenir. El terme DevOps és relativament nou i va ser popularitzat mitjançant *DevOps Open Days* (Bèlgica, 2009), i com a metodologia de desenvolupament ha anat guanyant adeptes des de grans a petites companyies, per a fer més i millors productes i serveis i, sobretot, a causa de diferents factors amb una forta presència avui dia com l'ús dels processos i metodologies de desenvolupament àgil, necessitat d'una major taxa de versions, àmplia disponibilitat d'entorns virtualitzats/*cloud*, major automatització de centres de dades i augment de les eines de gestió de configuració.

En aquest mòdul, es veuran diferents formes de crear i programar un sistema de còmput distribuït (clúster/*cloud*), les eines i biblioteques més importants per a complir aquest objectiu i els conceptes i eines vinculat a les metodologies DevOps.

Objectius

En els materials didàctics d'aquest mòdul, trobareu els continguts i les eines procedimentals per aconseguir els objectius següents:

1. Analitzar les diferents infraestructures i eines per al còmput d'altres prestacions (inclosa la virtualització) (HPC).
2. Configurar i instal·lar un clúster d'HPC i les eines de monitoratge corresponents.
3. Instal·lar i desenvolupar programes d'exemples en les principals API de programació: Posix Threads, OpenMPI, i OpenMP.
4. Instal·lar un clúster específic basat en una distribució *ad hoc* (Rocks).
5. Instal·lar una infraestructura per a prestar serveis en *cloud* (IaaS).
6. Eines DevOps per a automatitzar un centre de dades i gestions de la configuració.

1. Clusterització

çLa història dels sistemes informàtics és molt recent (es pot dir que comença en la dècada dels seixanta). Al principi, eren sistemes grans, pesats, cars, de pocs usuaris experts, no accessibles i lents. En la dècada dels setanta, l'evolució va permetre millores substancials dutes a terme per tasques interactives (*interactive jobs*), temps compartit (*time sharing*), terminals i amb una considerable reducció de la grandària. La dècada dels vuitanta es caracteritza per un augment notable de les prestacions (fins avui dia) i una reducció de la grandària en els anomenats *microordinadors*. La seva evolució ha estat a través de les estacions de treball (*workstations*) i els avenços en xarxes (LAN de 10 Mb/s i WAN de 56 kB/s el 1973 a LAN d'1/10 Gb/s i WAN amb ATM, *asynchronous transfer mode* d'1,2 Gb/s en l'actualitat o xarxes d'alt rendiment com Infiniband -96Gb/s- o Myrinet -10Gb/s-), que és un factor fonamental en les aplicacions multimèdia actuals i d'un futur proper. Els sistemes distribuïts, per la seva banda, van començar la seva història en la dècada dels setanta (sistemes de 4 o 8 ordinadors) i el seu salt a la popularitat el van fer en la dècada dels noranta. Si bé la seva administració, instal·lació i manteniment poden tenir una certa complexitat (cada vegada menys) perquè continuen creixent en grandària, les raons bàsiques de la seva popularitat són l'increment de prestacions que presenten en aplicacions intrínsecament distribuïdes (aplicacions que per la seva naturalesa són distribuïdes), la informació compartida per un conjunt d'usuaris, la compartició de recursos, l'alta tolerància a les fallades i la possibilitat d'expansió incremental (capacitat d'agregar més nodes per a augmentar les prestacions i de manera incremental). Un altre aspecte molt important en l'actualitat és la possibilitat, en aquesta evolució, de la virtualització. Les architectures cada vegada més eficients, amb sistemes *multicores*, han permès que la virtualització de sistemes es transformi en una realitat amb tots els avantatges (i possibles desavantatges) que això comporta.

1.1. Virtualització

La virtualització és una tècnica que està basada en l'abstracció dels recursos d'un ordinador, anomenada *Hypervisor* o VMM (*Virtual Machine Monitor*) que crea una capa de separació entre el maquinari de la màquina física (*host*) i el sistema operatiu de la màquina virtual (*virtual machine, guest*), i és un mitjà per a crear una "versió virtual" d'un dispositiu o recurs, com un servidor, un dispositiu d'emmagatzematge, una xarxa o fins i tot un sistema operatiu, on es divideix el recurs en un o més entorns d'execució. Aquesta capa de programari (VMM) maneja, gestiona i administra els quatre recursos principals d'un ordinador (CPU, memòria, xarxa i emmagatzematge) i els reparteix de mane-

ra dinàmica entre totes les màquines virtuals definides en l'ordinador central. D'aquesta manera, ens permet tenir diversos ordinadors virtuals executant-se sobre el mateix ordinador físic.

La màquina virtual, en general, és un sistema operatiu complet que s'executa com si estigués instal·lat en una plataforma de maquinari autònoma.

Enllaç d'interès

Per a saber més sobre virtualització, podeu visitar <http://en.wikipedia.org/wiki/Virtualization>. Es pot consultar una llista completa de programes de virtualització a http://en.wikipedia.org/wiki/Comparison_of_platform_virtual_machines.

1.1.1. Plataformes de virtualització

Els exemples més comuns de plataforma de virtualització amb llicències GPL o similars són Xen, KVM, Qemu (emulació), OpenVz, VirtualBox i Oracle VM (només el servidor); i entre les plataformes propietàries (algunes com *free to use*) VMware ESX/i, Virtual PC/Hyper-V, Parallels i Virtuozzo. Hi ha diferents taxonomies per a classificar la virtualització (i algunes plataformes poden executar-se en més d'una categoria), essent la més coneguda la següent:

1) Virtualització per HW: considerant com a *Full Virtualization* quan la màquina virtual simula tot el HW per a permetre a un sistema *guest* executar-se sense modificacions (sempre que estigui dissenyat per al mateix conjunt d'instruccions). Va ser la primera generació de VM per a processadors x86 i el que fa és una captura d'instruccions que accedeixen de manera prioritària a la CPU i les transforma en una crida a la VM perquè siguin emulades per programari. En aquest tipus de virtualització, poden executar-se Parallels, VirtualBox, Virtual Iron, Oracle VM, Virtual PC, Hyper-V, VMware per exemple.

2) Virtualització assistida per maquinari, on el maquinari dóna suport que facilita la construcció i treball de la VM i millora notablement les prestacions (a partir del 2005/6, Intel i AMD donen aquest suport com VT-x i AMD-V, respectivament). Els principals avantatges, respecte a altres formes de virtualització, és que no s'ha de tocar el sistema operatiu *guest* (com en paravirtualització) i s'obtenen millors prestacions, però com a contrapartida es necessita suport explícit en la CPU, la qual cosa no està disponible en tots els processadors x86/86_64. En aquest tipus de virtualització poden executar-se KVM, VMware Fusion, Hyper-V, Virtual PC, Xen, Parallels, Oracle VM, VirtualBox.

3) Paravirtualització: és una tècnica de virtualització en què la VM no necessàriament simula l'HW, sinó que presenta una API que pot ser utilitzada pel sistema *guest* (per la qual cosa, s'ha de tenir accés al codi font per a reemplaçar les crides d'un tipus per un altre). Aquest tipus de crides a aquesta API es denominen *hypercall* i Xen pot executar-se en aquesta forma.

4) Virtualització parcial: és el que es denomina *Address Space Virtualization*. La màquina virtual simula múltiples instàncies de l'entorn subjacent del maquinari (però no de tot), particularment l'*address space*. Aquest tipus de virtualit-

zació accepta compartir recursos i allotjar processos, però no permet instàncies separades de sistemes operatius *guest* i es troba en desús actualment.

5) Virtualització en un nivell del sistema operatiu: en aquest cas, la virtualització permet tenir múltiples instàncies aïllades i segures del servidor, totes executant-se sobre el mateix servidor físic i on el sistema operatiu *guest* coincidirà amb el sistema operatiu base del servidor (s'utilitza el mateix kernel). Plataformes dins d'aquest tipus de virtualització són FreeBSD jails (el pioner), OpenVZ, Linux-VServer, LXC, Virtuozzo.

La diferència entre instal·lar dos sistemes operatius i virtualitzar dos sistemes operatius és que en el primer cas tots els sistemes operatius que tinguem instal·lats funcionaran de la mateixa manera que si estiguessin instal·lats en diferents ordinadors, i necessitarem un gestor d'arrencada que en encendre l'ordinador ens permeti triar quin sistema operatiu volem utilitzar, però només en podrem tenir funcionant simultàniament un. En canvi, la virtualització permet executar moltes màquines virtuals amb els seus sistemes operatius i canviar de sistema operatiu com si es tractés de qualsevol altre programa; no obstant això, s'han de valorar molt bé les qüestions relacionades amb les prestacions, ja que si l'HW subjacent no és l'adequat, podrem notar moltes diferències en les prestacions entre el sistema operatiu instal·lat en base o el virtualitzat.

Entre els principals avantatges de la virtualització, hi ha:

- 1) Consolidació de servidors i millora de l'eficiència de la inversió en HW amb reutilització de la infraestructura existent.
- 2) Ràpid desplegament de nous serveis amb balanç dinàmic de càrrega i reducció dels sobredimensionaments de la infraestructura.
- 3) Increment d'*uptime* (temps que el sistema està al 100% en la prestació de serveis), increment de la tolerància a fallades (sempre que existeixi redundància física) i eliminació del temps de parada per a manteniment del sistema físic (migració de les màquines virtuals).
- 4) Manteniment a cost acceptable d'entorns de programari obsolets però necessaris per al negoci.
- 5) Facilitat de disseny i test de noves infraestructures i entorns de desenvolupament amb un baix impacte en els sistemes de producció i ràpida posada en marxa.
- 6) Millora de TCO (*Total Cost of Ownership*) i ROI (*Return on Investment*).
- 7) Menor consum d'energia que en servidors físics equivalents.

Com a desavantatges, podem esmentar:

- 1) Augmenta la probabilitat de fallades si no es considera redundància / alta disponibilitat (si es consoliden 10 servidors físics en un de potent equivalent, amb servidors virtualitzats, i deixen de funcionar tots els servidors en aquest, deixaran de prestar servei).
- 2) Rendiment inferior (possible) en funció de la tècnica de virtualització utilitzada i recursos disponibles.
- 3) Proliferació de serveis i màquines que incrementen les despeses d'administració/gestió (bàsicament, per l'efecte derivat de la facilitat de desplegament es tendeix a tenir-ne més dels necessaris).
- 4) Infraestructura desaprofitada (possible), ja que és habitual comprar una infraestructura major que la necessària en aquest moment per al possible creixement futur immediat.
- 5) Poden existir problemes de portabilitat, maquinari específic no suportat i compromís a llarg termini amb la infraestructura adquirida.
- 6) La presa de decisions en la selecció del sistema amfitrió pot ser complicada o condicionant.

Com és possible observar, els desavantatges es poden resoldre amb una planificació i presa de decisions adequada, i aquesta tecnologia és habitual en la prestació de serveis i totalment imprescindible en entorns de *cloud computing* (que veurem en aquest apartat també).

1.2. Beowulf

Beowulf [3, 1, 2, 4] és una arquitectura multiordenador que pot ser utilitzada per a aplicacions paral·leles/distribuïdes. El sistema consisteix bàsicament en un servidor i un o més clients connectats (generalment) a través d'Ethernet i sense la utilització de cap maquinari específic. Per a explotar aquesta capacitat de còmput, és necessari que els programadors tinguin un model de programació distribuït que, si bé és possible mitjançant UNIX (Sockets, RPC), pot implicar un esforç considerable, ja que són models de programació a nivell de *systems calls* i llenguatge C, per exemple; però aquesta forma de treball es pot considerar de baix nivell. Un model més avançat en aquesta línia de treball són els **Posix Threads**, que permeten explotar sistemes de memòria compartida i *multicores* de manera simple i fàcil. La capa de programari (interfície de programació d'aplicacions, API) aportada per sistemes com **Parallel Virtual Machine** (PVM) i **Message Passing Interface** (MPI) facilita notablement l'abstracció del sistema i permet programar aplicacions paral·leles/distribuïdes de manera senzilla i simple. Una de les formes bàsiques de treball és la de mestre-treballadors (*master-workers*), en què existeix un servidor (mestre) que distribueix la tasca que faran els treballadors. En grans sistemes (per exemple, de

1.024 nodes) hi ha més d'un mestre i nodes dedicats a tasques especials, com per exemple entrada/sortida o monitoratge. Una altra opció no menys interessant, sobretot amb l'auge de processadors *multicores*, és **OpenMP**, que és una API per a la programació multiprocés de memòria compartida. Aquesta capa de programari permet afegir concurrència als programes sobre la base del model d'execució *fork-join* i es compon d'un conjunt de directives de compilador, rutines de biblioteca i variables d'entorn, que influencien el comportament en temps d'execució i proporcionen als programadors una interfície simple i flexible per al desenvolupament d'aplicacions paral·leles i arquitectures de CPU que puguin executar més d'un fil d'execució simultani (*hyperthreading*) o que disposin de més d'un nucli per processador accedint a la mateixa memòria compartida.

Hi ha dos conceptes que poden donar lloc a dubtes i que són Cluster Beowulf i COW (*Cluster of Workstations*). Una de les principals diferències és que Beowulf "es veu" com una única màquina on s'accedeix als nodes remotament, ja que no disposen de terminal (ni de teclat), mentre que un COW és una agrupació d'ordinadors que poden ser utilitzats tant pels usuaris del COW com per altres usuaris en forma interactiva, a través de la pantalla i el teclat. Cal considerar que Beowulf no és un programari que transforma el codi de l'usuari en distribuït ni afecta el nucli del sistema operatiu (com per exemple, Mosix). Simplement, és una forma d'agrupació (un clúster) de màquines que executen GNU/Linux i actuen com un superordinador. Òbviament, hi ha una gran quantitat d'eines que permeten obtenir una configuració més fàcil, biblioteques o modificacions al nucli per a obtenir millors prestacions, però és possible construir un clúster Beowulf a partir d'un GNU/Linux estàndard i de programari convencional. La construcció d'un clúster Beowulf de dos nodes, per exemple, es pot dur a terme simplement amb les dues màquines connectades per Ethernet mitjançant un concentrador (*hub*), una distribució de GNU/Linux estàndard (Debian), el sistema d'arxius compartit (NFS) i tenint habilitats els serveis de xarxa, com per exemple `ssh`. En aquestes condicions, es pot argumentar que es disposa d'un clúster simple de dos nodes. En canvi, en un COW no necessitem tenir coneixements sobre l'arquitectura subjacent (CPU, xarxa) necessària per a fer una aplicació distribuïda en Beowulf però sí que és necessari tenir un sistema operatiu (per exemple, Mosix) que permeti aquest tipus de compartició de recursos i distribució de tasques (a més d'una API específica per a programar l'aplicació que és necessària en els dos sistemes). El primer Beowulf es va construir el 1994 i el 1998 comencen a aparèixer sistemes Beowulf/Linux en les llistes del Top500 (Avalon en la posició 314 amb 68 *cores* i 19,3 Gflops, juny del 98).

1.2.1. Com es configuren els nodes?

Primer de tot s'ha de modificar l'arxiu `/etc/hosts` perquè contingui la línia de `localhost` (amb 127.0.0.1) i la resta de noms a les IP internes dels restants nodes (aquest arxiu haurà de ser igual en tots els nodes). Per exemple:


```

127.0.0.1 localhost
192.168.0.254 lucix-server
I afegir les IP dels nodes (i per a tots els nodes), per exemple:
192.168.0.1 lucix1
192.168.0.2 lucix2
...
S'ha de crear un usuari (nteum) en tots els nodes, crear un grup i afegir aquest usuari al grup:
groupadd beowulf
adduser nteum beowulf
echo umask 007 >> /home/nteum/.bash_profile

```

Així, qualsevol arxiu creat per l'usuari nteum o qualsevol dins del grup serà modificable pel grup beowulf. S'ha de crear un servidor de NFS (i els altres nodes seran clients d'aquest NFS) i exportar el directori */home*, i així tots els clients veuran el directori *\$HOME* dels usuaris. Per a això, sobre el servidor s'edita el */etc/exports* i s'agrega una línia com

```
/home lucix*(rw, sync, no_root_esquaix)
```

Sobre cada node client es munta agregant en el */etc/fstab* perquè en l'arrencada el node munti el directori com a *192.168.0.254:/home /home nfs defaults 0 0*, també és aconsellable tenir en el servidor un directori */soft* (on s'instal·larà tot el programari perquè el vegin tots els nodes, i així ens evitem instal·lar aquest en cada node), i agregar-los a l'*export* com

```
/soft lucix*(rw, async, no_root_squash)
```

i en cada node agreguem en el *fstab* *192.168.0.254:/soft /soft nfs defaults 0 0*. També haurem de configurar el */etc/resolv.conf* i el *iptables* (per a fer un NAT) sobre el servidor si aquest té accés a Internet i volem que els nodes també hi tinguin accés (és molt útil sobretot a l'hora d'actualitzar el sistema operatiu).

A continuació, verifiquem que els serveis estan funcionant (tingueu en compte que l'ordre *chkconfig* pot no estar instal·lada en totes les distribucions) (en Debian, cal fer *apt-get install chkconfig*):

```

chkconfig --list sshd
chkconfig --list nfs

```

Que han d'estar a "on" en el nivell de treball que estiguem (generalment el nivell 3, però es pot esbrinar amb l'ordre *runlevel*) i, si no, s'hauran de fer les modificacions necessàries perquè aquests *daemons* s'iniciïn en aquest nivell (p. ex. *chkconfig name_service on; service name_service start*). Si bé estarem en un xarxa privada, per a treballar de manera segura és important treballar amb *ssh* i mai amb *rsh* o *rlogin*, per la qual cosa hauríem de generar les claus per a interconnectar en mode segur les màquines-usuari nteum sense *passwd*. Per a això, modifiquem (traiem el comentari #), de */etc/ssh/sshd_config* en les següents línies:

```

RSAAuthentication yes
AuthorizedKeysFile .ssh/authorized_keys

```

Reiniciem el servei (`service sshd restart`) i ens connectem amb l'usuari que desitgem i generem les claus:

```
ssh-keygen -t rsa
```

En el directori `$HOME/.ssh` s'hauran creat els arxius `id_rsa` i `id_rsa.pub` en referència a la clau privada i a la pública respectivament. Manualment, podem copiar `id_rsa.pub` en un arxiu anomenat `authorized_keys` en el mateix directori (ja que en estar compartit per NFS, serà el mateix directori que veuran tots els nodes i verifiquem els permisos: 600 per al directori `.ssh`, `authorized_keys` i `id_rsa` i 644 per a `id_rsa.pub`). És convenient també instal·lar NIS sobre el clúster, i així evitem haver de definir cada usuari en els nodes i un servidor DHCP per a definir tots els paràmetres de xarxa que cada node sol·liciti durant l'etapa de *boot*. Per a això, consulteu l'apartat de servidors i el de xarxa de l'assignatura Administració GNU/Linux, on s'expliquen amb detall aquestes configuracions.

A partir d'aquí, ja tenim un clúster Beowulf per a executar aplicacions amb interfície a MPI (com es veurà en els següents subapartats) per a aplicacions distribuïdes (també poden ser aplicacions amb interfície a PVM, però és recomanable MPI per qüestions d'eficiència i prestacions). Hi ha sobre diferents distribucions (Debian, FC incloses) l'aplicació `system-config-cluster`, que permet configurar un clúster sobre la base d'una eina gràfica o `clusterssh` o `dish`, que permeten gestionar i executar ordres en un conjunt de màquines de manera simultània (ordres útils quan necessitem fer operacions sobre tots els nodes del clúster, per exemple, apagar-los, reiniciar-los o fer còpies d'un arxiu a tots aquests).

Enllaç d'interès

Informació adicional:
<https://wiki.debian.org/highperformancecomputing>

1.3. Beneficis del còmput distribuït

Quins són els beneficis del còmput en paral·lel? Veurem això amb un exemple [3]. Considerem un programa per a sumar nombres (per exemple, $4+5+6+\dots$) anomenat `sumdis.c`:

```

#include <stdio.h>

int main (int argc, char** argv){
long inicial, final, resultat, tmp;
    if (argc < 2) {
        printf (" Ús: %s Núm. inicial Núm. final\n",argv[0]);
        return (4); }
    else {
        inicial = atoll(argv[1]);
        final = atoll(argv[2]);
        resultat = 0;}
    for (tmp = inicial; tmp <= final; tmp++){resultat += tmp; };
}

```

```
    printf("%lu\n", resultat);
    return 0;
}
```

Ho compilem amb `gcc -o sumdis sumdis.c` i si mirem l'execució d'aquest programa, per exemple, amb

```
time ./sumdis 1 1000000
500000500000

real 0m0.005s
user 0m0.000s
sys   0m0.004s
```

es podrà observar que el temps en una màquina Debian 7.5 sobre una màquina virtual (Virtualbox) amb processador i7 és de 5 mil·lèsimes de segon. Si, en canvi, fem des d'1 a 16 milions, el temps real puja fins a 0,050 s, és a dir, 10 vegades més, per la qual cosa, si es consideren 1.600 milions, el temps serà de prop de 4 segons.

La idea bàsica del còmput distribuït és repartir el treball. Així, si disposem d'un clúster de quatre màquines (lucix1–lucix4) amb un servidor i on l'arxiu executable es comparteix per NFS, és interessant dividir l'execució mitjançant `ssh` de manera que el primer sumi d'1 a 400.000.000, el segon, de 400.000.001 a 800.000.000, el tercer, de 800.000.001 a 1.200.000.000, i el quart, d'1.200.000.001 a 1.600.000.000. Les següents ordres mostren una possibilitat. Considerem que el sistema té el directori `/home` compartit per NFS i l'usuari **adminp**, que té adequadament configurades les claus per a executar el codi sense *password* sobre els nodes, executarà:

```
mkfifo out1      Crea una cua fifo en /home/adminp
./distr.sh & time cat out1 | awk '{total += $1 } END {printf "%lf", total}'
```

S'executa l'ordre `distr.sh`; es recol·lecten els resultats i se sumen mentre es mesura el temps d'execució. El *shell script* `distr.sh` pot ser una cosa com:

```
ssh lucix1 /home/nteum/sumdis 1 400000000 > /home/nteum/out1 < /dev/null &
ssh lucix2 /home/nteum/sumdis 400000001 800000000 > /home/nteum/out1 < /dev/null &
ssh lucix3 /home/nteum/sumdis 800000001 1200000000 > /home/nteum/out1 < /dev/null &
ssh lucix4 /home/nteum/sumdis 1200000001 1600000000 > /home/nteum/out1 < /dev/null &
```

Podrem observar que el temps es redueix notablement (aproximadament en un valor proper a 4) i no exactament de manera lineal, però molt propera. Òbviament, aquest exemple és molt simple i només vàlid per a finalitats demostratives. Els programadors utilitzen biblioteques que els permeten portar a terme el temps d'execució, la creació i comunicació de processos en un sistema distribuït (per exemple MPI o OpenMP).

1.3.1. Com cal programar per a aprofitar la concurrència?

Hi ha diverses maneres d'expressar la concurrència en un programa. Les tres més comunes són les següents:

- 1) Utilitzant fils (o processos) en el mateix processador (multiprogramació amb superposició del còmput i l'E/S).
- 2) Utilitzant fils (o processos) en sistemes *multicore*.
- 3) Utilitzant processos en diferents processadors que es comuniquen mitjançant missatges (MPS, *Message Passing System*).

Aquests mètodes poden ser implementats sobre diferents configuracions de maquinari (memòria compartida o missatges) i, si bé tots dos mètodes tenen els seus avantatges i desavantatges, els principals problemes de la memòria compartida són les limitacions en l'escalabilitat (ja que tots els *cores*/processadors utilitzen la mateixa memòria i el nombre d'aquests en el sistema està limitat per l'amplada de banda de la memòria) i, en els sistemes de pas de missatges, la latència i velocitat dels missatges a la xarxa. El programador haurà d'avaluar quin tipus de prestacions necessita, les característiques de l'aplicació subjacent i el problema que es desitja solucionar. No obstant això, amb els avenços de les tecnologies de *multicores* i de xarxa, aquests sistemes han crescut en popularitat (i en quantitat). Les API més comunes avui dia són Posix Threads i OpenMP per a memòria compartida i MPI (en les seves versions OpenMPI o Mpich) per a pas de missatges. Com hem esmentat anteriorment, hi ha una altra biblioteca molt difosa per a passos de missatges, anomenada PVM, però la versatilitat i prestacions que s'obtenen amb MPI l'ha deixat relegada a aplicacions petites o per a aprendre a programar en sistemes distribuïts. Aquestes biblioteques, a més, no limiten la possibilitat d'utilitzar fils (encara que en un nivell local) i tenir concurrència entre processament i entrada/sortida.

Per a portar a terme una aplicació paral·lela/distribuïda, es pot partir de la versió sèrie o mirar l'estructura física del problema i determinar quines parts poden ser concurrents (independents). Les parts concurrents seran candidates a reescriure's com a codi paral·lel. A més, s'ha de considerar si és possible reemplaçar les funcions algebraïques per les seves versions paral·lelitzades (per exemple, ScaLapack *Scalable Linear Algebra Package* (es poden provar en Debian els diferents programes de prova que es troben en el paquet scalapack-mpi-test, per exemple i directament, instal·lar les biblioteques libscalapack-mpi-dev). També és convenient esbrinar si hi ha alguna aplicació similar paral·lela que pugui orientar-nos sobre la mode de construcció de l'aplicació paral·lel*.

*<http://www.mpich.org/>

Paral·lelitzar un programa no és una tasca fàcil, ja que s'ha de tenir en compte la **lleï d'Amdahl**, que afirma que l'increment de velocitat (*speedup*) està limitat per la fracció de codi (f), que pot ser paral·lelitzat de la següent manera:

$$\text{speedup} = \frac{1}{1-f}$$

Aquesta llei implica que amb una aplicació seqüencial $f = 0$ i l'*speedup* = 1, mentre que amb tot el codi paral·lel $f = 1$ i l'*speedup* es fa infinit. Si considerem valors possibles, un 90% ($f = 0,9$) del codi paral·lel significa un *speedup* igual a 10, però amb $f = 0,99$ l'*speedup* és igual a 100. Aquesta limitació es pot evitar amb algorismes escalables i diferents models de programació d'aplicació (paradigmes):

- 1) Mestre-treballador: el mestre inicia a tots els treballadors i coordina el seu treball i el d'entrada/sortida.
- 2) *Single Process Multiple Data* (SPMD): el mateix programa que s'executa amb diferents conjunts de dades.
- 3) Funcional: diversos programes que porten a terme una funció diferent en l'aplicació.

En resum, podem concloure:

- 1) Proliferaió de màquines multitasca (multiusuari) connectades per xarxa amb serveis distribuïts (NFS i NIS).
- 2) Són sistemes heterogenis amb sistemes operatius de tipus NOS (*Networked Operating System*), que ofereixen una sèrie de serveis distribuïts i remots.
- 3) La programació d'aplicacions distribuïdes es pot efectuar en diferents nivells:
 - a) Utilitzant un model client-servidor i programant a baix nivell (*sockets*) o fent servir memòria compartida a baix nivell (Posix Threads).
 - b) El mateix model, però amb API d'"alt" nivell (OpenMP, MPI).
 - c) Utilitzant altres models de programació com, per exemple, programació orientada a objectes distribuïts (RMI, CORBA, Agents, etc.).

1.4. Memòria compartida. Models de fils (*threading*)

Normalment, en una arquitectura client-servidor, els clients sol·liciten als servidors determinats serveis i esperen que aquests els contestin amb la major eficàcia possible. Per a sistemes distribuïts amb servidors amb una càrrega molt

alta (per exemple, sistemes d'arxius de xarxa, bases de dades centralitzades o distribuïdes), el disseny del servidor es converteix en una qüestió crítica per a determinar el rendiment general del sistema distribuït. Un aspecte crucial en aquest sentit és trobar la manera òptima de manejar l'E/S, tenint en compte el tipus de servei que ofereix, el temps de resposta esperat i la càrrega de clients. No hi ha un disseny predeterminat per a cada servei, i escollir el correcte dependrà dels objectius i restriccions del servei i de les necessitats dels clients.

Les preguntes que hem de contestar abans de triar un determinat disseny són les següents: quant temps es triga en un procés de sol·licitud del client?, quantes d'aquestes sol·licituds és probable que arribin durant aquest temps?, quant temps pot esperar el client?, quant afecta aquesta càrrega del servidor les prestacions del sistema distribuït? A més, amb l'avenç de la tecnologia de processadors ens trobem que disposem de sistemes *multicore* (múltiples nuclis d'execució) que poden executar seccions de codi independents. Si es dissenyen els programes en forma de múltiples seqüències d'execució i el sistema operatiu ho suporta (i GNU/Linux és un d'aquests), l'execució dels programes es reduirà notablement i s'incrementaran en forma (gairebé) lineal les prestacions en funció dels *cores* de l'arquitectura.

1.4.1. Multifils (*multithreading*)

Segons les últimes tecnologies en programació per a aquest tipus d'aplicacions (i així ho demostra l'experiència), els dissenys més adequats són aquells que utilitzen models de multifils (*multithreading models*), en els quals el servidor té una organització interna de processos paral·lels o fils cooperants i concurrents.

Un fil (*thread*) és una seqüència d'execució (fil d'execució) d'un programa, és a dir, diferents parts o rutines d'un programa que s'executen concurrentment en un únic processador i accediran a les dades compartides al mateix temps.

Quins avantatges aporta això respecte a un programa seqüencial? Considerem que un programa té tres rutines A, B i C. En un programa seqüencial, la rutina C no s'executarà fins que s'hagin executat A i B. Si, en canvi, A, B i C són fils, les tres rutines s'executaran concurrentment i, si en aquestes hi ha E/S, tindrem concurrència d'execució amb E/S del mateix programa (procés), cosa que millorarà notablement les prestacions d'aquest programa. Generalment, els fils estan continguts dins d'un procés, i diferents fils d'un mateix procés poden compartir alguns recursos, mentre que diferents processos no. L'execució de múltiples fils en paral·lel necessita el suport del sistema operatiu i en els processadors moderns hi ha optimitzacions del processador per a suportar models multifil (*multithreading*) a més de les architectures *multicores* on hi ha múltiples nuclis i on cadascun pot executar un *thread*.

Generalment, hi ha quatre models de disseny per fils (en ordre de complexitat creixent):

- 1) **Un fil i un client:** en aquest cas, el servidor entra en un bucle sense fi, escoltant per un port i davant la petició d'un client s'executen els serveis en el mateix fil. Altres clients hauran d'esperar que acabi el primer. És fàcil d'implementar però només atén un client alhora.
- 2) **Un fil i diversos clients amb selecció:** en aquest cas, el servidor utilitza un sol fil, però pot acceptar múltiples clients i multiplexar el temps de CPU entre aquests. Es necessita una gestió més complexa dels punts de comunicació (*sockets*), però permet crear serveis més eficients, encara que presenta problemes quan els serveis necessiten una alta càrrega de CPU.
- 3) **Un fil per client:** és, probablement, el model més popular. El servidor espera per peticions i crea un fil de servei per a atendre cada nova petició dels clients. Això genera simplicitat en el servei i una alta disponibilitat, però el sistema no escala amb el nombre de clients i pot saturar el sistema molt ràpidament, ja que el temps de CPU dedicat davant una gran càrrega de clients es redueix notablement i la gestió del sistema operatiu pot ser molt complexa.
- 4) **Servidor amb fils en granja (*worker threads*):** aquest mètode és més complex però millora l'escalabilitat dels anteriors. Hi ha un nombre fix de fils treballadors (*workers*) als quals el fil principal distribueix el treball dels clients. El problema d'aquest mètode és l'elecció del nombre de treballadors: amb un nombre elevat, cauran les prestacions del sistema per saturació; amb un nombre massa baix, el servei serà deficient (els clients hauran d'esperar). Normalment, serà necessari sintonitzar l'aplicació per a treballar amb un determinat entorn distribuït.

Hi ha diferents formes d'expressar en un nivell de programació amb fils: paral·lisme en un nivell de tasques o paral·lisme a través de les dades. Triar el model adequat minimitza el temps necessari per a modificar, depurar i sintonitzar el codi. La solució a aquesta disjuntiva és descriure l'aplicació en termes de dos models basats en un treball en concret:

- Tasques paral·leles amb fils independents que poden atendre tasques independents de l'aplicació. Aquestes tasques independents seran encapsulades en fils que s'executaran asincrònicament i s'hauran d'utilitzar biblioteques com Posix Threads (Linux/Unix) o Win32 Thread API (Windows), que han estat dissenyades per a suportar concurrència en un nivell de tasca.
- Model de dades paral·leles per a calcular llaços intensius; és a dir, la mateixa operació ha de repetir-se un nombre elevat de vegades (per exemple, comparar una paraula amb les paraules d'un diccionari). Per a aquest cas, és possible encarregar la tasca al compilador de l'aplicació o, si no és possible, que el programador descriu el paral·lisme utilitzant l'en-

torn OpenMP, que és una API que permet escriure aplicacions eficients sota aquest tipus de models.

Una aplicació d'informació personal (*Personal Information Manager*) és un bon exemple d'una aplicació que conté concurrència en un àmbit de tasques (per exemple, accés a la base de dades, llibreta d'adreces, calendari, etc.). Això podria ser en pseudocodi:

```
Function addressBook;
Function inBox;
Function calendar;
Program PIM      {
    CreateThread (addressBook);
    CreateThread (inBox);
    CreateThread (calendar); }
```

Podem observar que hi ha tres execucions concurrents sense relació. Un altre exemple d'operacions amb paral·lelisme de dades podria ser un corrector d'ortografia, que en pseudocodi seria: `Function SpellCheck {loop (word = 1, words_in_file) compareToDictionary (word);}`

S'ha de tenir en compte que tots dos models (fils paral·lels i dades paral·leles) poden existir en una mateixa aplicació. A continuació, es mostrarà el codi d'un productor de dades i un consumidor de dades basat en Posix Threads. Per a compilar sobre Linux, per exemple, s'ha d'utilitzar `gcc -o pc pc.c -lpthread`.

```
#include <pthread.h>
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#define QUEUE_SIZE 10
#define LOOP 20

void *producer (void *args);
void *consumer (void *args);
typedef struct { /* Estructura del buffer compartit i descriptors de threads */
int buf[QUEUE_SIZE]; long head, tail; int full, empty;
pthread_mutex_t *mut; pthread_cond_t *notFull, *notEmpty;
} queue;

queue *queueInit (void); /* Prototip de funció: inicialització del buffer */
void queueDelete (queue *q); /* Prototip de funció: esborrament del buffer */
void queueAdd (queue *q, int in); /* Prototip de funció: inserir element en el buffer */
void queueDel (queue *q, int *out); /* Prototip de funció: treure element del buffer */

int main () {
queue *fifo; pthread_t pro, con; fifo = queueInit ();
if (fifo == NULL) { fprintf (stderr, " Error en crear buffer.\n"); exit (1); }
pthread_create (&pro, NULL, producer, fifo); /* Creació del thread productor */
pthread_create (&con, NULL, consumer, fifo); /* Creació del thread consumidor */
pthread_join (pro, NULL); /* main () espera fins que s'acabin els dos threads */
pthread_join (con, NULL);
queueDelete (fifo); /* Eliminació del buffer compartit */
return 0; } /* Fi */

void *producer (void *q) { /*Funció del productor */
queue *fifo; int i;
fifo = (queue *)q;
for (i = 0; i < LOOP; i++) { /* Inserir en el buffer elements=LOOP */
pthread_mutex_lock (fifo->mut); /* Semàfor per a entrar a inserir */
```



```

while (fifo->full) {
printf ("Productor: queue FULL.\n");
pthread_cond_wait (fifo->notFull, fifo->mut); }
/* Bloqueig del productor si el buffer està ple, i allibera el semàfor mut
perquè hi pugui entrar el consumidor. Continuarà quan el consumidor executi
pthread_cond_signal (fifo->notFull);*/
queueAdd (fifo, i); /* Insert element en el buffer */
pthread_mutex_unlock (fifo->mut); /* Allibero el semàfor */
pthread_cond_signal (fifo->notEmpty); /* Desbloqueig consumidor si està bloquejat */
usleep (100000); /* Dormo 100 mseg per permetre que el consumidor s'activi */
}
return (NULL); }

void *consumer (void *q) { /*Funció del consumidor */
queue *fifo; int i, d;
fifo = (queue *)q;
for (i = 0; i < LOOP; i++) { /* Traiem del buffer elements=LOOP*/
pthread_mutex_lock (fifo->mut); /* Semàfor per a entrar a treure */
while (fifo->empty) {
printf (" Consumidor: queue EMPTY.\n");
pthread_cond_wait (fifo->notEmpty, fifo->mut); }
/* Bloqueig del consumidor si el buffer està buit, alliberant el semàfor mut
perquè pugui entrar el productor. Continuarà quan el consumidor executi
pthread_cond_signal (fifo->notEmpty);*/
queueDel (fifo, &d); /* Trec element del buffer */
pthread_mutex_unlock (fifo->mut); /* Allibero el semàfor */
pthread_cond_signal (fifo->notFull); /* Desbloquejo productor si està bloquejat */
printf (" Consumidor: Rebut %d.\n", d);
usleep(200000); /* Dormo 200 mseg per a permetre que el productor s'activi */
}
return (NULL); }

queue *queueInit (void) {
queue *q;
q = (queue *)malloc (sizeof (queue)); /* Creació del buffer */
if (q == NULL) return (NULL);
q->empty = 1; q->full = 0; q->head = 0; q->tail = 0;
q->mut = (pthread_mutex_t *) malloc (sizeof (pthread_mutex_t));
pthread_mutex_init (q->mut, NULL); /* Creació del semàfor */
q->notFull = (pthread_cond_t *) malloc (sizeof (pthread_cond_t));
pthread_cond_init (q->notFull, NULL); /* Creació de la variable condicional notFull*/
q->notEmpty = (pthread_cond_t *) malloc (sizeof (pthread_cond_t));
pthread_cond_init (q->notEmpty, NULL); /* Creació de la variable condicional notEmpty*/
return (q); }

void queueDelete (queue *q) {
pthread_mutex_destroy (q->mut); free (q->mut);
pthread_cond_destroy (q->notFull); free (q->notFull);
pthread_cond_destroy (q->notEmpty); free (q->notEmpty);
free (q); }

void queueAdd (queue *q, int in) {
q->buf[q->tail] = in; q->tail++;
if (q->tail == QUEUE_SIZE) q->tail = 0;
if (q->tail == q->head) q->full = 1;
q->empty = 0;
return; }

void queueDel (queue *q, int *out){
*out = q->buf[q->head]; q->head++;
if (q->head == QUEUE_SIZE) q->head = 0;
if (q->head == q->tail) q->empty = 1;
q->full = 0;
return; }

```

1.5. OpenMP

L'OpenMP (*Open-Multi Processing*) és una interfície de programació d'aplicacions (API) amb suport multiplataforma per a la programació en C/C++ i Fortran

de processos amb ús de memòria compartida sobre plataformes Linux/Unix (i també Windows). Aquesta infraestructura es compon d'un conjunt de directives del compilador, rutines de la biblioteca i variables d'entorn que permeten aprofitar recursos compartits en memòria i en temps d'execució. Definit conjuntament per un grup dels principals fabricants de maquinari i programari, OpenMP permet utilitzar un model escalable i portàtil de programació, que proporciona als usuaris una interfície simple i flexible per al desenvolupament, sobre plataformes paral·leles, d'aplicacions d'escriptori fins a aplicacions d'altres prestacions sobre superordinadors. Una aplicació construïda amb el model híbrid de la programació paral·lela pot executar-se en un ordinador utilitzant tant OpenMP com Message Passing Interface (MPI) [5].

OpenMP és una implementació multifil, mitjançant la qual un fil mestre divideix la tasques sobre un conjunt de fils treballadors. Aquests fils s'executen de manera simultània i l'entorn d'execució porta a terme l'assignació d'aquests als diferents processadors de l'arquitectura. La secció del codi que està dissenyada per a funcionar en paral·lel està marcada amb una directiva de preprocessament que crearà els fils abans que la secció s'executi. Cada fil tindrà un identificador (*id*) que s'obtéindrà a partir d'una funció (en C/C++ `omp_get_thread_num()`) i, després de l'execució paral·lela, els fils s'uniran de nou en la seva execució sobre el fil mestre, que continuarà amb l'execució del programa. Per defecte, cada fil executarà una secció paral·lela de codi independent però es poden declarar seccions de "treball compartit" per a dividir una tasca entre els fils, de manera que cada fil executi part del codi assignat. D'aquesta manera, és possible tenir en un programa OpenMP paral·lelisme de dades i paral·lelisme de tasques convivint.

Els principals elements d'OpenMP són les sentències per a la creació de fils, la distribució de càrrega de treball, la gestió de dades d'entorn, la sincronització de fils i les rutines en un nivell d'usuari. OpenMP utilitza en C/C++ les directives de preprocessament conegudes com a *pragma* (`#pragma omp <resta del pragma>`) per a diferents construccions. Així, per exemple, `omp parallel` s'utilitza per a crear fils addicionals per a executar el treball indicat en la sentència paral·lela on el procés original és el fil mestre (`id=0`). El conegut programa que imprimeix "Hello, world" utilitzant OpenMP i multifils és*:

```
#include <stdio.h>
#include <omp.h>

int main(int argc, char* argv[]){
    #pragma omp parallel
    printf("Hello, world.\n");
    return 0;}
```

On executarà un *thread* per a cada *core* disponible en l'arquitectura. Per a especificar *work-sharing constructs*, s'utilitza:

- **omp for** o **omp do**: reparteix les iteracions d'un llaç en múltiples fils.
- **sections**: assigna blocs de codi independents consecutius a diferents fils.

***Compileu amb `gcc -fopenmp -o hello hello.c` (en algunes distribucions -Debian està instal·lat per defecte- s'ha de tenir instal·lada la biblioteca GCC OpenMP (GOMP) `apt-get install libgomp1`.**

- **single:** especifica que el bloc de codi serà executat per un sol fil amb una sincronització (*barrier*) al final.
- **master:** similar a `single`, però el codi del bloc serà executat pel fil mestre i no hi ha *barrier* implicat al final.

Per exemple, per a inicialitzar els valors d'un *array* en paral·lel utilitzant fils per a fer una porció del treball (compileu, per exemple, amb `gcc -fopenmp -o init2 init2.c`):

```
#include <stdio.h>
#include <omp.h>
#define N 1000000
int main(int argc, char *argv[]) {
    float a[N]; long i;
    #pragma omp parallel for
    for (i=0; i<N; i++) a[i] = 2*i;
    return 0;
}
```

Si executem amb el *pragma* i després el comentem i calculem el temps d'execució (`time ./init2`), veiem que el temps d'execució passa de 0.003 s a 0.007 s, la qual cosa mostra la utilització del *dualcore* del processador.

Atès que OpenMP és un model de memòria compartida, moltes variables en el codi són visibles per a tots els fils per defecte. Tanmateix, de vegades és necessari tenir variables privades i passar valors entre blocs seqüencials del codi i blocs paral·lels, per la qual cosa és necessari definir atributs a les dades (*data clauses*) que permetin diferents situacions.

- **shared:** les dades a la regió paral·lela són compartides i accessibles per tots els fils de manera simultània.
- **private:** les dades a la regió paral·lela són privades per a cada fil, i cadascun en té una còpia sobre una variable temporal.
- **default:** permet al programador definir com seran les dades dins de la regió paral·lela (`shared`, `private` o `none`).

Un altre aspecte interessant d'OpenMP són les directives de sincronització:

- **critical section:** el codi emmarcat serà executat per fils, però només un per vegada (no hi haurà execució simultània) i es manté l'exclusió mútua.
- **atomic:** similar a `critical section`, però avisa el compilador perquè faci servir instruccions de maquinari especials de sincronització i així obtenir millors prestacions.
- **ordered:** el bloc és executat en l'ordre d'un programa seqüencial.
- **barrier:** cada fil espera que els restants hagin acabat la seva execució (implica sincronització de tots els fils al final del codi).
- **nowait:** especifica que els fils que acabin el treball assignat poden continuar.

A més, OpenMP proveeix de sentències per a la planificació (*scheduling*) del tipus `schedule (type, chunk)` (on el tipus pot ser `static`, `dynamic` o `guided`) o proporciona control sobre les sentències `if`, la qual cosa permetrà definir si es paral·lelitzava o no en funció de si l'expressió és veritable o no. OpenMP també proporciona un conjunt de funcions de biblioteca, com per exemple:

- **omp_set_num_threads**: defineix el nombre de fils que cal fer servir a la següent regió paral·lela.
- **omp_get_num_threads**: obté el nombre de fils que s'estan usant en una regió paral·lela.
- **omp_get_max_threads**: obté la màxima quantitat possible de fils.
- **omp_get_thread_num**: retorna el número del fil.
- **omp_get_num_procs**: retorna el màxim nombre de processadors que es poden assignar al programa.
- **omp_in_parallel**: retorna un valor diferent de zero si s'executa dins d'una regió paral·lela.

Veurem a continuació alguns exemples simples (compileu amb la instrucció `gcc -fopenmp -o out_file input_file.c`):

```
/* Programa simple multithreading amb OpenMP */
#include <omp.h>
int main() {
    int iam = 0, np = 1;

    #pragma omp parallel private(iam, np)
    #if defined (_OPENMP)
        np = omp_get_num_threads();
        iam = omp_get_thread_num();
    #endif

    printf("Hello from thread %d out of %d \n", iam, np);
}

/* Programa simple amb threads imbricats amb OpenMP */
#include <omp.h>
#include <stdio.h>
main(){
    int x=0, nt, tid, ris;

    omp_set_nested(2);
    ris=omp_get_nested();
    if (ris) printf("Paral·lelisme imbricat actiu %d\n", ris);
    omp_set_num_threads(25);
    #pragma omp parallel private (nt, tid) {
        tid = omp_get_thread_num();
        printf("Thread %d\n", tid);
        nt = omp_get_num_threads();
        if (omp_get_thread_num() == 1)
            printf("Nombre de Threads: %d\n", nt);
    }
}

/* Programa simple d'integració amb OpenMP */
#include <omp.h>
#include <stdio.h>
#define N 100
main() {
    double local, pi=0.0, w; long i;
    w = 1.0 / N;
```

```

#pragma omp parallel private(i, local)
{
    #pragma omp single
    pi = 0.0;
    #pragma omp for reduction(+: pi)
    for (i = 0; i < N; i++) {
        local = (i + 0.5)*w;
        pi = pi + 4.0/(1.0 + local*local);
        printf ("Pi: %f\n",pi);
    }
}

/* Programa simple de reducció amb OpenMP */
#include <omp.h>
#include <stdio.h>
#define NUM_THREADS 2
main () {
    int i; double ZZ, res=0.0;
    omp_set_num_threads(NUM_THREADS);
    #pragma omp parallel for reduction(+:res) private(ZZ)
    for (i=0; i< 1000; i++) {
        ZZ = i*i;
        res = res + ZZ;
        printf("ZZ: %f, res: %f\n", ZZ, res);
    }
}

```

Nota

Es recomana veure també els exemples que es poden trobar en la referència [6].

1.6. MPI, Message Passing Interface

La definició de l'API de MPI [9, 10] ha estat el treball resultant del MPI Forum (MPIF), que és un consorci de més de 40 organitzacions. MPI té influències de diferents arquitectures, llenguatges i treballs al món del paral·lelisme, com per exemple: WRC (Ibm), Intel NX/2, Express, nCUBE, Vertex, p4, Parmac i contribucions de ZipCode, Chimp, PVM, Chamaleon, PICL.

El principal objectiu de MPIF va ser dissenyar una API, sense relació particular amb cap compilador ni biblioteca, que permetés la comunicació eficient (*memory-to-memory copy*), còmput i comunicació concurrent i descàrrega de comunicació, sempre que hi hagués un coprocessador de comunicacions. A més, es demanava que suportés el desenvolupament en ambients heterogenis, amb interfície C i F77 (incloent C++, F90), on la comunicació fos fiable i les fallades, resoltes pel sistema. L'API també havia de tenir interfície per a diferents entorns, disposar d'una implementació adaptable a diferents plataformes amb canvis insignificants i que no interferís amb el sistema operatiu (*thread-safety*). Aquesta API va ser dissenyada especialment per a programadors que utilitzessin el *Message Passing Paradigm* (MPP) en C i F77, per a aprofitar la seva característica més rellevant: la portabilitat. El MPP es pot executar sobre màquines multiprocessador, xarxes d'estacions de treball i fins i tot sobre màquines de memòria compartida. La primera versió de l'estàndard va ser MPI-1 (que si bé hi ha molts desenvolupaments sobre aquesta versió, es considera en EOL), la versió MPI-2 va incorporar un conjunt de millores, com ara creació de processos dinàmics, *one-sided communication*, entrada/sortida paral·lela entre d'altres, i finalment l'última versió, MPI-3 (considerada com una revisió major), inclou *nonblocking collective operations*, *one-sided operations* i suport per a Fortran 2008.[7]

Molts aspectes han estat dissenyats per a aprofitar els avantatges del maquinari de comunicacions sobre SPC (*scalable parallel computers*), i l'estàndard ha estat acceptat de manera majoritària pels fabricants de maquinari en paral·lel i distribuït (SGI, SUN, Cray, HPConvex, IBM, etc.). Hi ha versions lliures (per exemple, Mpich, LAM/MPI i OpenMPI) que són totalment compatibles amb les implementacions comercials efectuades pels fabricants de maquinari i inclouen comunicacions punt a punt, operacions col·lectives i grups de processos, context de comunicacions i topologia, suport per a F77 i C, i un entorn de control, administració i *profiling*. [11, 12, 10]

Tanmateix, hi ha també alguns punts que poden presentar alguns problemes en determinades arquitectures, com són la memòria compartida, l'execució remota, les eines de construcció de programes, la depuració, el control de fils, l'administració de tasques i les funcions d'entrada/sortida concurrents (la major part d'aquests problemes de falta d'eines estan resolts a partir de la versió 2 de l'API –MPI2). Un dels problemes de MPI1, ja que no té creació dinàmica de processos, és que només suporta models de programació MIMD (*Multiple Instruction Multiple Data*) i comunicant-se via les anomenades MPI. A partir de MPI-2 i amb els avantatges de la creació dinàmica de processos, ja es poden implementar diferents paradigmes de programació com ara *master-worker/farmer-tasks*, *divide & conquer*, paral·lelisme especulatiu, etc. (o almenys sense tanta complexitat i major eficiència en la utilització dels recursos).

Per a la instal·lació de MPI es recomana utilitzar la distribució (en alguns casos la compilació pot ser complexa a causa de les dependències d'altres paquets que pot necessitar). Debian inclou la versió OpenMPI (sobre Debian 7.5 és versió 2, però es poden baixar les fonts i compilar-les) i Mpich2 (Mpich3 disponible en <http://www.mpich.org/downloads/>). La millor elecció serà OpenMPI, ja que combina les tecnologies i els recursos d'altres projectes diversos (FT-MPI, LA-MPI, LAM/MPI i PACX-MPI) i suporta totalment l'estàndard MPI-2 (i des de la versió 1.75, suporta la versió MPI3). Entre altres característiques d'OpenMPI tenim: és conforme a MPI-2/3, *thread safety & concurrency*, creació dinàmica de processos, alt rendiment i gestió de treballs tolerants a fallades, instrumentació en temps d'execució, *job schedulers*, etc. Per a això s'han d'instal·lar els paquets `openmpi-dev`, `openmpi-bin`, `openmpi-common` i `openmpi-doc`. A més, Debian 7.5 inclou una altra implementació de MPI anomenada LAM (paquets `lam*`). S'ha de considerar que si bé les implementacions són equivalents des del punt de vista de MPI, tenen diferències quant a la gestió i procediments de compilació/execució/gestió.

1.6.1. Configuració d'un conjunt de màquines per a fer un clúster adaptat a OpenMPI

Per a la configuració d'un conjunt de màquines per a fer un clúster adaptat a OpenMPI [14], s'han de seguir els següents passos:

- 1) Cal tenir les màquines “visibles” (per exemple, mitjançant un `ping`) a través de TCP/IP (IP pública/privada).
- 2) És recomanable que totes les màquines tinguin la mateixa arquitectura de processador, així és més fàcil distribuir el codi, amb versions similars de Linux (si pot ser, amb el mateix tipus de distribució).
- 3) Es recomana tenir NIS o, si no, s’ha de generar un mateix usuari (per exemple, `mpiuser`) a totes les màquines i el mateix directori `$HOME` muntat per NFS.
- 4) Nosaltres anomenarem les màquines `slave1`, `slave2`, etc. (ja que després resultarà més fàcil fer les configuracions), però es poden anomenar les màquines com cadascú prefereixi.
- 5) Una de les màquines serà el mestre i les restants, `slaveX`.
- 6) S’ha d’instal·lar en tots els nodes (suposem que tenim la distribució Debian): `openmpi-bin`, `openmpi-common`, `openmpi-dev`. Cal verificar que en totes les distribucions es treballa amb la mateixa versió d’OpenMPI.
- 7) En Debian, els executables estan en `/usr/bin` però si estan en un *path* diferent, haurà d’agregar-se a `mpiuser` i també verificar que `LD_LIBRARY_PATH` apunta a `/usr/lib`.
- 8) En cada node esclau ha d’instal·lar-se el *SSH server* (instal·leu el paquet `openssh-server`), i sobre el mestre, el client (paquet `openssh-client`).
- 9) S’han de crear les claus públiques i privades fent `ssh-keygen -t dsa` i copiar a cada node amb `ssh-copy-id` per a aquest usuari (només s’ha de fer en un node, ja que, com tindrem el directori `$HOME` compartit per NFS per a tots els nodes, amb una còpia n’hi ha prou).
- 10) Si no es comparteix el directori, cal assegurar que cada esclau coneix que l’usuari `mpiuser` es pot connectar sense *passwd*, per exemple fent: `ssh slave1`.
- 11) S’ha de configurar la llista de les màquines sobre les quals s’executarà el programa, per exemple `/home/mpiuser/.mpi_hostfile` i amb el següent contingut:

```
# The Hostfile for Open MPI
# The master node, slots=2 is used because it is a dual-processor machine.
  localhost slots=2
# The following slave nodes are single processor machines:
  slave1
  slave2
  slave3
```

- 12) OpenMPI permet utilitzar diferents llenguatges, però aquí utilitzarem C. Per a això, cal executar sobre el mestre `mpicc testprogram.c`. Si es desitja veure què incorpora `mpicc`, es pot fer `mpicc -showme`.
- 13) Per a executar en local, podríem fer `mpirun -np 2 ./myprogram` i per a executar sobre els nodes remots (per exemple 5 processos), `mpirun -np 2 -hostfile ./mpi_hostfile ./myprogram`.

És important notar que `np` és el nombre de processos o processadors en què s'executarà el programa i es pot posar el nombre que es desitgi, ja que OpenMPI intentarà distribuir els processos de manera equilibrada entre totes les màquines. Si hi ha més processos que processadors, OpenMPI/Mpich utilitzarà les característiques d'intercanvi de tasques de GNU/Linux per a simular l'execució paral·lela. A continuació, es veuran dos exemples: `Srtest` és un programa simple per a establir comunicacions entre processos punt a punt, i `cpi` calcula el valor del nombre π de manera distribuïda (per integració).

```
/* Srtest Program */
#include "mpi.h"
#include <stdio.h>
#include <string.h>
#define BUFLLEN 512

int main(int argc, char *argv[]){
    int myid, numprocs, next, namelen;
    char buffer[BUFLLEN], processor_name[MPI_MAX_PROCESSOR_NAME];
    MPI_Status status;
    MPI_Init(&argc,&argv); /* Ha de posar-se abans d'altres crides MPI, sempre */
    MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
    MPI_Comm_rank(MPI_COMM_WORLD,&myid); /*Integra el procés en un grup de comunicacions*/
    MPI_Get_processor_name(processor_name,&namelen); /*Obté el nom del processador*/

    fprintf(stderr,"Procés %d sobre %s\n", myid, processor_name);
    fprintf(stderr,"Procés %d de %d\n", myid, numprocs);
    strcpy(buffer,"hello there");
    if (myid == numprocs-1) next = 0;
    else next = myid+1;

    if (myid == 0) { /*Si és l'inicial, envia string de buffer*/
        printf("%d sending '%s' \n",myid,buffer);fflush(stdout);
        MPI_Send(buffer, strlen(buffer)+1, MPI_CHAR, next, 99, MPI_COMM_WORLD);
        /*Blocking Send, 1:buffer, 2:size, 3:tipus, 4:destinació, 5:tag, 6:context*/
        printf("%d receiving \n",myid);fflush(stdout);
        MPI_Recv(buffer, BUFLLEN, MPI_CHAR, MPI_ANY_SOURCE, 99, MPI_COMM_WORLD,&status);
        printf("%d received '%s' \n",myid,buffer);fflush(stdout);
        /* mpdprintf(001,"%d receiving \n",myid); */
    }
    else {
        printf("%d receiving \n",myid);fflush(stdout);
        MPI_Recv(buffer, BUFLLEN, MPI_CHAR, MPI_ANY_SOURCE, 99, MPI_COMM_WORLD,&status);
        /* Blocking Recv, 1:buffer, 2:size, 3:tipus, 4:font, 5:tag, 6:context, 7:status*/
        printf("%d received '%s' \n",myid,buffer);fflush(stdout);
        /* mpdprintf(001,"%d receiving \n",myid); */
        MPI_Send(buffer, strlen(buffer)+1, MPI_CHAR, next, 99, MPI_COMM_WORLD);
        printf("%d sent '%s' \n",myid,buffer);fflush(stdout);
    }
    MPI_Barrier(MPI_COMM_WORLD); /*Sincronitza tots els processos*/
    MPI_Finalize(); /*Allibera els recursos i acaba*/
    return (0);
}

/* CPI Program */
#include "mpi.h"
#include <stdio.h>
#include <math.h>
double f( double );
double f( double a) { return (4.0 / (1.0 + a*a)); }
int main( int argc, char *argv[] ) {
    int done = 0, n, myid, numprocs, i;
    double PI25DT = 3.141592653589793238462643;
    double mypi, pi, h, sum, x; double startwtime = 0.0, endwtime;
    int namelen; char processor_name[MPI_MAX_PROCESSOR_NAME];
    MPI_Init(&argc,&argv);
```



```

MPI_Comm_size(MPI_COMM_WORLD, &numprocs); /*Indica el nombre de processos en el grup*/
MPI_Comm_rank(MPI_COMM_WORLD, &myid); /*Id del procés*/
MPI_Get_processor_name(processor_name, &namelen); /*Nom del procés*/
fprintf(stderr, "Procés %d sobre %s\n", myid, processor_name);
n = 0;
while (!done) {
    if (myid == 0) { /*Si és el primer...*/
        if (n == 0) n = 100; else n = 0;
        startwtime = MPI_Wtime(); /* Time Clock */
        MPI_Bcast(&n, 1, MPI_INT, 0, MPI_COMM_WORLD); /*Broadcast a la resta*/
        /*Envia des del 4 arg. a tots els processos del grup. Els restants que no són 0
        copiaran el buffer des de 4 o arg -procés 0-*/
        /*1:buffer, 2:size, 3:tipus, 5:grup */
        if (n == 0) done = 1;
        else {h = 1.0 / (double) n;
            sum = 0.0;
            for (i = myid + 1; i <= n; i += numprocs) {
                x = h * ((double)i - 0.5); sum += f(x); }
            mypi = h * sum;
            MPI_Reduce(&mypi, &pi, 1, MPI_DOUBLE, MPI_SUM, 0, MPI_COMM_WORLD);
            /* Combina els elements del Send Buffer de cada procés del grup usant
            l'operació MPI_SUM i retorna el resultat en el Recv Buffer. Ha de ser cridada
            per tots els processos del grup fent servir els mateixos arguments*/

            /*1:sendbuffer, 2:recvbuffer, 3:size, 4:tipus, 5:oper, 6:root, 7:context*/
            if (myid == 0) { /*només el P0 imprimeix el resultat*/
                printf("Pi és aproximadament %.16f, l'error és %.16f\n", pi, fabs(pi - PI25DT));
                endwtime = MPI_Wtime();
                printf("Temps d'execució = %f\n", endwtime-startwtime); }
        }
    }
    MPI_Finalize(); /*Allibera recursos i acaba*/
    return 0;
}

```

Per a visualitzar l'execució d'un codi paral·lel/distribuït en MPI, hi ha una aplicació anomenada XMPI (en Debian xmpi) que permet “veure”, durant l'execució, l'estat de l'execució de les aplicacions sobre MPI, però està vinculada al paquet LAM/MPI. Per a visualitzar i analitzar codi sobre OpenMPI, s'hauria de (i és recomanable) baixar i compilar el codi de MPE* o TAU**, que són eines de *profiling* molt potents i que no requereixen gaire treball ni dedicació per a engegar-les.

*<http://www.mcs.anl.gov/research/projects/perfvis/software/mpe/>
 **<http://www.cs.uoregon.edu/research/tau/home.php>

1.7. Rocks Cluster

Rocks Cluster és una distribució de Linux per a clústers d'ordinadors d'alt rendiment. Les versions actuals de Rocks Cluster estan basades en CentOS (CentOS 6.5 el juliol del 2014) i, com a instal·lador, Anaconda amb certes modificacions, que simplifica la instal·lació “en massa” en molts ordinadors. Rocks Cluster inclou moltes eines (com ara MPI) que no formen part de CentOS però són els components que transformen un grup d'ordinadors en un clúster. Les instal·lacions poden personalitzar-se amb paquets de programari addicionals anomenats *rolls*. Els *rolls* estenen el sistema integrant automàticament els mecanismes de gestió i empaquetament usats pel programari bàsic, i simplifiquen àmpliament la instal·lació i configuració d'un gran nombre d'ordinadors. S'ha creat una gran quantitat de *rolls*, com per exemple SGE *roll*, Córndor *roll*, Xen *roll*, el Java *roll*, Ganglia *roll*, etc.*** Rocks Cluster és una dis-

***http://www.rocksclusters.org/wordpress/?page_id=4

tribució molt emprada en l'àmbit de clústers, per la seva facilitat d'instal·lació i incorporació de nous nodes i per la gran quantitat de programes per al manteniment i monitoratge del clúster.

Les principals característiques de Rocks Cluster són:

- 1) Facilitat d'instal·lació, ja que només és necessari completar la instal·lació d'un node anomenat *node mestre* (*frontend*), la resta s'instal·la amb Avalache, que és un programa P2P que ho fa de manera automàtica i evita haver d'instal·lar els nodes un a un.
- 2) Disponibilitat de programes (conjunt molt ampli de programes que no és necessari compilar i transportar) i facilitat de manteniment (només s'ha de mantenir el node mestre).
- 3) Disseny modular i eficient, pensat per a minimitzar el trànsit de xarxa i utilitzar el disc dur propi de cada node per a només compartir la informació mínima i imprescindible.

La instal·lació es pot seguir pas a pas des del lloc web de la distribució [15] i els autors garanteixen que no es triga més d'una hora per a una instal·lació bàsica. Rocks Cluster permet, en l'etapa d'instal·lació, diferents mòduls de programari, els *rolls*, que contenen tot el necessari per a fer la instal·lació i la configuració de sistema amb aquestes noves addicions de manera automàtica i, a més, decidir en el *frontend* com serà la instal·lació en els nodes esclaus, quins *rolls* estaran actius i quina arquitectura s'usarà. Per al manteniment, inclou un sistema de còpia de seguretat de l'estat, anomenat *Roll Restore*. Aquest *roll* guarda els arxius de configuració i *scripts* (i fins i tot, es poden afegir els arxius que es vulguin).

1.7.1. Guia ràpida d'instal·lació

Aquest és un resum breu de la instal·lació proposada en [15] per a la versió 6.1 i es parteix de la base que el *frontend* té (mínim) 30 GB de disc dur, 1 GB de RAM, arquitectura x86-64 i 2 interfícies de xarxa (eth0 per a la comunicació amb Internet i eth1 per a la xarxa interna); per als nodes 30 GB de disc dur, 512 MB de RAM i 1 interfície de xarxa (xarxa interna). Després d'obtenir els discos *Kernel/Boot Roll*, *Base Roll*, *US Roll CD1/2* (o en defecte d'això, DVD equivalent), inserim *kernel boot* i seguim els següents passos:

- 1) Arrenquem el *frontend* i veurem una pantalla en la qual introduïm *build* i ens preguntarà la configuració de la xarxa (IPv4 o IPv6).
- 2) El següent pas que s'ha de fer és seleccionar els *rolls* (per exemple, seleccionant els "CD/DVD-based Roll" i anem introduint els següents *rolls*) i marcar en les successives pantalles quins són els que volem.

Enllaços d'interès

Per a una llista detallada de les eines incloses en Rocks Cluster, consulteu <http://www.rocksclusters.org/roll-documentation/base/5.5/>.

Enllaços d'interès

És possible descarregar els discos des de: http://www.rocksclusters.org/wordpress/?page_id=80

3) La següent pantalla ens demanarà informació sobre el clúster (és important definir bé el nom de la màquina, *Fully-Qualified Host Name*, ja que en cas contrari fallarà la connexió amb diferents serveis) i també informació per a la xarxa privada que connectarà el *frontend* amb els nodes i la xarxa pública (per exemple, la que connectarà el *frontend* amb Internet) així com DNS i passarel·les.

4) A continuació, se sol·licitarà la contrasenya per al *root*, la configuració del servei de temps i la partició del disc (es recomana escollir “auto”).

5) Després de formatar els discos, sol·licitarà els CD de *rolls* indicats i instal·larà els paquets i farà un *reboot* del *frontend*.

6) Per a instal·lar els nodes, s’ha d’entrar com a *root* en el *frontend* i executar `insert-ethers`, que capturarà les peticions de DHCP dels nodes i els agregarà a la base de dades del *frontend*, i seleccionar l’opció “*Compute*” (per defecte, consulteu la documentació per a les altres opcions).

7) Posem en marxa el primer node i en el *boot order* de la BIOS generalment es tindrà CD, PXE (Network Boot), Hard Disk (si l’ordinador no suporta PXE, llavors feu el *boot* del node amb el *Kernel Roll CD*). En el *frontend* es veurà la petició i el sistema l’agregarà com a `compute-0-0` i començarà la descàrrega i instal·lació dels arxius. Si la instal·lació falla, s’hauran de reiniciar els serveis `httpd`, `mysqld` i `autofs` en el *frontend*.

8) A partir d’aquest punt, es pot monitorar la instal·lació executant la instrucció `rocks-console`, per exemple amb `rocks-console compute-0-0`. Després que s’hagi instal·lat tot, es pot sortir de `insert-ethers` prement la tecla F8. Si es disposa de dos *racks*, després d’instal·lat el primer es pot començar el segon fent `insert-ethers -cabinet=1`, els quals rebran els noms `compute-1-0`, `compute-1-1`, etc.

9) A partir de la informació generada en consola per `rocks list host`, generarem la informació per a l’arxiu `machines.conf`, que tindrà un aspecte com:

```
nteum slot=2
compute-0-0 slots=2
compute-0-1 slots=2
compute-0-2 slots=2
compute-0-3 slots=2
```

i que després haurem d’executar amb

```
mpirun -np 10 -hostfile ./machines.conf ./mpi_program_to_execute.
```

1.8. FAI

FAI és una eina d’instal·lació automatitzada per a desplegar Linux en un clúster o en un conjunt de màquines en forma desatesa. És equivalent a *Kickstart*

de RH, o Alice de SuSE. FAI pot instal·lar Debian, Ubuntu i RPM de distribucions Linux. Els seus avantatges són que mitjançant aquesta eina es pot desplegar Linux sobre un node (físic o virtual) simplement fent que arrenqui per PXE i quedi preparat per a treballar i totalment configurat (quan es diu un, es podria dir 100 amb el mateix esforç per part de l'administrador) i sense cap interacció pel mig. Per tant, és un mètode escalable per a la instal·lació i actualització d'un clúster Beowulf o una xarxa d'estacions de treball sense supervisió i amb poc esforç. FAI utilitza la distribució Debian, una col·lecció d'*scripts* (la majoria en Perl) i *cfengine* i *Preseeding d-i* per al procés d'instal·lació i canvis en els arxius de configuració.

És una eina pensada per a administradors de sistemes que han d'instal·lar Debian en desenes o centenars d'ordinadors i pot utilitzar-se a més com a eina d'instal·lació de propòsit general per a instal·lar (o actualitzar) un clúster de còmput HPC, de servidors web, o un *pool* de bases de dades i configurar la gestió de la xarxa i els recursos en forma desatesa. Aquesta eina permet tractar (mitjançant un concepte que incorpora anomenat *classes*) amb un maquinari diferent i diferents requisits d'instal·lació en funció de la màquina i característiques de què es disposi en la seva preconfiguració, la qual cosa permet fer desplegaments massius eficients i sense intervenció d'administrador (una vegada que l'eina hagi estat configurada de manera adequada). [29, 30, 31]

1.8.1. Guia ràpida d'instal·lació

En aquest apartat veurem una guia de la instal·lació bàsica sobre màquines virtuals (en Virtualbox) per a desplegar altres màquines virtuals, però es pot adaptar/ampliar a les necessitats de l'entorn amb mínimes intervencions i és una eina que no té *daemons*/DB i només consisteix en *scripts* (consulteu les referències indicades).

Instal·lació del servidor. Si bé hi ha un paquet anomenat *fai-quickstart* és millor fer-ho per parts, ja que molts dels paquets ja estan instal·lats, i si no, es poden instal·lar quan és necessari (vegeu més informació en http://fai-project.org/fai-guide/_anchor_id_inst_xreflabel_inst_installing_fai.html).

Descàrrega dels paquets: en Debian Wheezy hi ha tots els paquets. És millor baixar-se l'última versió de <http://fai-project.org/download/wheezy/> i encara que s'han de descarregar tots els paquets *fai-alguna-cosa-.deb* només utilitzarem *fai-server* i *fai-doc* en aquest exemple.

Instal·lació de la clau (*key*):

```
wget -O - http://fai-project.org/download/074BCDE4.asc | sudo  
apt-key add -
```

Instal·lació dels paquets: dins del directori on es trobin i com a *root* cal executar

```
dkpg -i fai-server_x.y_all.deb; dkpg -i fai-doc_x.y_all.deb
```

reemplaçant la "x.y" per a la versió descarregada.

Configurar el DHCP del servidor: el *boot* dels nodes i la transferència del sistema operatiu l'efectuarem per una petició PXE i qui primer ha d'actuar és el *dhcp* per a assignar al node els paràmetres de xarxa i l'arxiu de *boot*, per la qual cosa haurem de modificar */etc/dhcp/dhcpd.conf* agregant les primitives **next-server** i **filename** a les zones de la nostra xarxa:

```
...
authoritative;
...
subnet 192.168.168.0 netmask 255.255.255.0 {
    ...
    next-server 192.168.168.254;
    filename "fai/pxelinux.0";
    ...
}
```

Configuració del servei TFTP: la transferència dels arxius del SO es farà pel servei TFTP (*Trivial File Transfer Protocol*), per la qual cosa haurem de comprovar si el *tftp-hpa* està instal·lat i ben configurat amb `netstat -anp|grep :69`. Haurà de donar una cosa similar a `udp 0 0 0.0.0.0:69 0.0.0.0:*`. Si no, caldrà verificar si està instal·lat (`dpkg -l | grep tftp`) i, en cas que no estigui instal·lat, executar `apt-get install tftpd-hpa`. A continuació, verificarem que la configuració és l'adequada */etc/default/tftpd-hpa* (s'ha de recordar reiniciar el servei si es modifica):

```
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/srv/tftp"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure"
```

Crear la configuració de FAI: executant com a *root* (els arxius de configuració es trobaran en */srv/fai/config* i hi ha regles per a modificar-los –vegeu la documentació):

```
fai-setup
echo '/srv/fai/config 192.168.168.0/24(async,ro,no_subtree_check,no_root_squash)' >> /etc/exports
/etc/init.d/nfs-kernel-server restart
```

```
Amb un exportfs, veurem una cosa com:
/home          192.168.168.0/24
...
/srv/fai/nfsroot 192.168.168.0/24
/srv/fai/config 192.168.168.0/24
```

Generar una configuració FAI-client: per a tots els clients (després, podrem fer configuracions diferents):

```
fai-chboot -IBv -u nfs://192.168.168.254/srv/fai/config default
```

Haurem de modificar l'arxiu generat perquè treballi amb NFSv3; amb la versió del kernel utilitzat en Wheezy (3.2.x) només podem utilitzar V3 (si es desitja utilitzar NFSv4, haurem de canviar a un kernel 3.12.x que estan disponibles en repositoris com BackPorts <http://backports.debian.org/>). L'arxiu `/srv/tftp/fai/pxelinux.cfg/default` ha de quedar com (s'ha modificat la variable *root*):

```
default fai-generated

label fai-generated
kernel vmlinuz-3.2.0-4-amd64
append initrd=initrd.img-3.2.0-4-amd64 ip=dhcp root=192.168.168.254:/srv/fai/nfsroot:vers=3
aufs FAI_FLAGS=verbose,sshd,reboot FAI_CONFIG_SRC=nfs://192.168.168.254/srv/fai/config FAI_ACTION=install
```

Copiar els mínims fitxers de configuració:

```
cp -a /usr/share/doc/fai-doc/examples/simple/* /srv/fai/config/
```

Configuració del node: la nostra prova de concepte la farem per a un node en VirtualBox i primer és necessari instal·lar les expansions de la versió que tenim instal·lada. Per exemple, si tenim VirtualBox 4.3.10, haurem d'anar a <https://www.virtualbox.org/wiki/downloads> i instal·lar *VirtualBox 4.3.10 Oracle VM VirtualBox Extension Pack All supported platforms* (això ens permetrà tenir un dispositiu que faci *boot* per a PXE) i podrem verificar que estan instal·lades en el menú de VirtualBox – >File – >Preferences – >Extensions. Després haurem de crear una nova màquina virtual amb un disc buit i seleccionar-la en *System* l'opció de *boot order network* en primer lloc. Arrencarem la màquina i veurem que rep IP (verifiquem els possibles errors en `/var/log/messages|syslog`) i que comença baixant el **initrd-img** i després el **kernel**.

Els detalls de la configuració es poden consultar en http://fai-project.org/fai-guide/_anchor_id_config_xreflabel_config_installation_details.html

1.9. Logs

Linux manté un conjunt de registres anomenats *system logs* o *logs* simplement, que permeten analitzar què ha passat i quan en el sistema, a partir dels esdeveniments que recull del propi *kernel* o de les diferents aplicacions i gairebé totes les distribucions es troben en `/var/log`. Probablement, els dos arxius més importants (i que amb més o menys presència estan en totes les distribucions) són `/var/log/messages` i `/var/log/syslog`, els que tenen registres (ASCII) d'esdeveniments com ara errors del sistema, (re)iniciis/apagats, errors d'aplicacions,

advertiments, etc. Existeix una ordre, `dmesg`, que permet a més veure els missatges d'inici (en algunes distribucions són els que apareixen en la consola o amb la tecla Esc en la manera gràfica d'arrencada, o Ctrl+F8 en altres distribucions) per a visualitzar els passos seguits durant l'arrencada (tenint en compte que pot ser molt extens, es recomana executar `dmesg | more`).

Per la qual cosa, el sistema ens permetrà analitzar què ha passat i esbrinar les causes que han produït aquest registre i on/quant. El sistema inclòs en Debian és `rsyslog` (<http://www.rsyslog.com/>) amb un servei (a través del *daemon* **rsyslogd**) que reemplaça a l'original `syslog`. És un sistema molt potent i versàtil i la seva configuració és través de l'arxiu `/etc/rsyslog.conf` o arxius en el directori `/etc/rsyslog.d`. Podeu mirar la documentació sobre la seva configuració (`man rsyslog.conf` o a la pàgina indicada), però en resum inclou una sèrie de directives, filtres, *templates* i regles que permeten canviar el comportament dels *logs* del sistema. Per exemple, les regles són de tipus **recurs.nivell acció** però es pot combinar més d'un recurs separat per ',', o diferents nivells, o tots '*' o negar-ho '!', i si posem '=' davant del nivell, indica només aquest nivell i no tots els superiors, que és el que ocorre quan només s'indica un nivell (els nivells poden ser de menor a major importància *debug, info, notice, warning, err, crit, alert, emerg*). Per exemple, `*.warning /var/log/messages` indicarà que tots els missatges de *warning,err,crit,alert,emerg* de qualsevol recurs vagin a parar a aquest arxiu. Es pot incloure també un '-' davant del nom del fitxer, que indica que no se sincronitzi el fitxer després de cada escriptura per a millorar les prestacions del sistema i reduir la càrrega.

Un aspecte interessant dels *logs* és que els podem centralitzar des de les diferents màquines de la nostra infraestructura en un determinat servidor, per a no haver de connectar-nos si volem "veure" què està ocorrent en els *logs* en aquestes màquines (sobretot si el nombre de màquines és elevat). Per a això, en cada client hem de modificar l'arxiu `/etc/rsyslog.conf` (on la IP serà la del servidor que recollirà els *logs*):

```
# Provides TCP forwarding.
*. * @192.168.168.254:514
```

En el servidor, haurem de treure el comentari (#) dels mòduls de recepció per TCP en `/etc/rsyslog.conf`:

```
# Provides TCP syslog reception
$ModLoad imtcp
$InputTCPServerRun 514
```

Després de fer les modificacions de cada arxiu, és important no oblidar fer un `/etc/init.d/rsyslog restart`. A partir d'aquest moment, podem mirar el `/var/log/syslog` del servidor, per exemple, i veurem els registres marcats amb el nom (o IP) de la màquina on s'ha generat el *log*.

Existeix un paquet anomenat `syslog-ng` que presenta característiques avançades (per ex., xifratge o filtres basats en continguts) o altres d'equivalents amb `rsyslog`. Trobareu més informació en <http://www.balabit.com/network-security/syslog-ng>.

1.9.1. Octopussy

Octopussy és una eina gràfica que permet analitzar els *logs* de diferents màquines i s'integra amb `syslog` o equivalents reemplaçant-los. Entre les seves principals característiques, suporta LDAP, alertes per correu, missatges IM (Jabber), s'integra amb Nagios i Zabbix, permet generar reports i enviar-los per correu, FTP i scp, fer mapes de l'arquitectura per a mostrar estats i incorpora una gran quantitat de serveis predefinits per a registrar els *logs*.^[32]

La seva instal·lació (que pot resultar una mica complicada) comença per afegir el repositori *non-free* `/etc/apt/sources.list` per a descarregar paquets addicionals (`libmail-sender-perl`) que necessitarà aquest programari:

```
deb http://ftp.es.debian.org/debian/ wheezy non-free
deb-src http://ftp.es.debian.org/debian/ wheezy non-free
```

Després, haurem d'executar `apt-get update`.

Posteriorment, haurem de descarregar el paquet Debian des de l'adreça web <http://sourceforge.net/projects/syslog-analyzer/files/> (per exemple el paquet `octopussy_x.i.x_all.deb` on la `x.y.z` és la versió `-10.0.14` el juliol de 2014) i instal·lar-lo.

```
dpkg -i octopussy_1.0.x_all.deb
apt-get -f install
```

Després haurem de modificar l'arxiu `etc/rsyslog.conf`:

```
$ModLoad imuxsock # provides support for local system logging
$ModLoad imklog    # provides kernel logging support (previously done by rklogd)
#$ModLoad immark   # provides --MARK-- message capability
# provides UDP syslog reception
$ModLoad imudp
$UDPServerRun 514
# provides TCP syslog reception
$ModLoad imtcp
$InputTCPServerRun 514
```

I reiniciar el `rsyslog` `/etc/init.d/rsyslog restart`.

Després s'hauran de generar els certificats per a *Octopussy Web Server* (també es pot utilitzar l'OpenCA per a generar els certificats web propis, tal com es va explicar en l'apartat de servidors):


```
openssl genrsa > /etc/octopussy/server.key  
openssl req -new -x509 -nodes -sha1 -days 365 -key /etc/octopussy/server.key > /etc/octopussy/server.crt
```

En l'últim, no s'ha d'introduir *passwd* i cal recordar en *Common Name* incloure el nom.domini de la nostra màquina.

Configurar l'arxiu de la configuració d'Apache */etc/octopussy/apache2.conf* pel valor correcte:

```
SSLCertificateFile /etc/octopussy/server.crt  
SSLCertificateKeyFile /etc/octopussy/server.key
```

Reiniciar el servidor (propi) de web que després es connectarà amb el nostre Apache per SSL i port 8888.

```
/etc/init.d/octopussy web-stop  
/etc/init.d/octopussy web-start
```

I comprovar que funciona en l'URL <https://sysdw.nteum.org:8888/index.asp>, acceptar el certificat i introduir com a Usuari/Passwd admin/admin. A partir de la interfície general, s'haurà de configurar els *Devices*, *Service*, *Alerts*, etc. per a definir el comportament que volem que tingui l'aplicació.

Si volem deshabilitar i tornar al *rsyslog* original (i donat que té serveis propis que s'engeguen durant l'arrencada), haurem d'executar *update-rc.d octopussy remove* i renombrar els arxius de configuració en */etc/rsyslog.d* de l'aplicació (per exemple,

```
mv /etc/rsyslog.d/xyz-ocotopussy.conf /etc/rsyslog.d/xyz-octopussy.conf.org
```

reemplaçant xyz pel que correspongui) i reiniciant *rsyslog* amb */etc/init.d/rsylog restart*

1.9.2. Eines de monitoratge addicionals

A més de les eines que s'han vist en l'apartat de monitoratge (com Ganglia, Nagios|Icinga, MRTG o Zabbix), hi ha altres eines que poden ajudar en la gestió i administració d'un clúster com són Cacti, XyMon i Collectd (també disponibles en Debian). En aquest apartat, dedicarem una petita referència a Cacti i XYMon, ja que per la seva visió o capacitat d'integració permeten tenir informació a l'instant sobre com estan ocorrent les execucions en el clúster.

Cacti. Cacti [16] és una solució per a la visualització d'estadístiques de xarxa i va ser dissenyada per a aprofitar el poder d'emmagatzematge i la funcionalitat de generar gràfiques que posseeix RRDtool (similar a Ganglia). Aquesta eina,

desenvolupada en PHP, proveeix de diferents formes de visualització, gràfics avançats i disposa d'una interfície d'usuari fàcil d'usar, que la fan interessant tant per a xarxes LAN com per a xarxes complexes amb centenars de dispositius.

La seva instal·lació és simple i requereix prèviament tenir instal·lat MySQL. Després, fent `apt-get install cacti` s'instal·larà el paquet i al final ens demanarà si volem instal·lar la interfície amb la base de dades, i ens sol·licitarà el nom d'usuari i contrasenya d'aquesta i la contrasenya de l'usuari admin de la interfície de Cacti (si no li hem donat, contindrà la que defineix per defecte, que és usuari admin i `passwd admin`). A continuació instal·larà la configuració en el lloc d'Apache2 (`/etc/apache2/conf.d/cacti.conf`) i tornarà a arrencar els servidors i podrem connectar-nos a l'URL <http://sysdw.nteum.org/cacti/>. La primera tasca que cal portar a terme és fer els ajustos pertinents de la instal·lació (ens mostrarà una *checklist*), després s'ha de canviar la contrasenya i ens mostrarà a continuació una pantalla amb les pestanyes de gràfics i la de consola, en la qual trobarem les diferents opcions per a configurar i engegar el monitoratge més adequat per al nostre lloc. Seguint el procediment indicat en http://docs.cacti.net/manual:088:2_basics.1_first_graph, serà molt fàcil incorporar els gràfics més adequats per a monitorar la nostra instal·lació. És interessant incorporar a Cacti un *plugin* anomenat *weathermap* (<http://www.networkweathermap.com/>) que ens permetrà veure un diagrama de la infraestructura en temps real i els elements monitorats amb dades dinàmiques sobre el seu estat*.

*<http://www.networkweathermap.com/manual/0.97b/pages/cacti-plugin.html>

Xymon. Xymon [17] és un monitor de xarxa/serveis (sota llicència GPL) que s'executa sobre màquines GNU/Linux. L'aplicació es va inspirar en la versió *opensource* de l'eina Big Brother i es va dir Hobbit però, com que aquesta era una marca registrada, finalment va canviar a Xymon. Xymon ofereix un monitoratge gràfic de les màquines i serveis amb una visualització adequada i jeràrquica òptima per a quan es necessita tenir en una sola visualització l'estat de la infraestructura (i sobretot quan s'han de monitorar centenars de nodes). També es pot entrar en els detalls i mirar qüestions específiques dels nodes, gràfics i estadístiques, però entrant en nivells interiors de detalls. El monitoratge dels nodes requereix un client que serà l'encarregat d'enviar la informació al *host* (equivalent a NRPE en Nagios). La seva instal·lació en Debian és simple: `apt-get install xymon`. Després, podrem modificar l'arxiu `/etc/hobbit/bb-host` per a agregar una línia com a *group server* 158.109.65.67 sysdw.nteum.org ssh <http://sysdw.nteum.org> i modificar el fitxer `/etc/apache2/conf.d/hobbit` per a canviar la línia `Allow from localhost` per `Allow from all`, reiniciar els dos serveis, `service hobbit restart`, i `service apache2 restart` i ho tindrem disponible en l'URL: <http://sysdw.nteum.org/hobbit/> (no us descuideu la barra final). És molt fàcil agregar nous *hosts* (es declaren en l'arxiu anterior), i instal·lar el client en els nodes tampoc té cap dificultat.

2. Cloud

Les infraestructures *cloud* es poden classificar en 3 + 1 grans grups en funció dels serveis que presten i a qui els presten:

1) Públiques: els serveis es troben en servidors externs i les aplicacions dels clients es barregen en els servidors i amb altres infraestructures. L'avantatge més clar és la capacitat de processament i emmagatzematge sense instal·lar màquines localment (no hi ha inversió inicial ni manteniment) i es paga per ús. Té un retorn de la inversió ràpid i pot resultar difícil integrar aquests serveis amb altres de propis.

2) Privades: les plataformes es troben dins de les instal·lacions de l'empresa/institució i són una bona opció per a aquells que necessiten una alta protecció de dades (aquests continuen dins de la pròpia empresa). És més fàcil integrar aquests serveis amb altres de propis, però hi ha inversió inicial en infraestructura física, sistemes de virtualització, amplada de banda, seguretat i despesa de manteniment, la qual cosa implica un retorn més lent de la inversió. No obstant això, entre tenir infraestructura i tenir-ne *cloud* i virtualitzada, aquesta última és millor pels avantatges de major eficiència, millor gestió i control i major aïllament entre projectes i rapidesa en la provisió.

3) Híbrides: combinen els models de núvols públics i privats i permeten mantenir el control de les aplicacions principals a la vegada que s'aprofita el *cloud computing* en els llocs on tingui sentit amb una inversió inicial moderada i, alhora, permet comptar amb els serveis que es necessitin sota demanda.

4) Comunitat: canalitzen necessitats agrupades i les sinergies d'un conjunt o sector d'empreses per a oferir serveis de *cloud* a aquests grups d'usuaris.

A més, hi ha diferents capes sota les quals es poden contractar aquests serveis:

1) *Infrastructure as a Service* (IaaS): es contracta capacitat de procés i d'emmagatzematge que permeten desplegar aplicacions pròpies que per motius d'inversió, infraestructura, cost o falta de coneixements no es volen instal·lar en la pròpia empresa (exemples d'aquest tipus són EC2/S3 d'Amazon i Azure de Microsoft).

2) *Platform as a Service* (PaaS): es proporciona a més un servidor d'aplicacions (on s'executaran les nostres aplicacions) i una base de dades, on es podran instal·lar les aplicacions i executar-les, les quals s'hauran de desenvolupar d'acord amb unes indicacions del proveïdor (per exemple, Google App Engine).

3) *Programari as a Service* (SaaS): comunament s'identifica amb *cloud*, on l'usuari final paga un lloguer per a l'ús de programari sense adquirir-lo en propietat, instal·lar-lo, configurar-lo i mantenir-lo (en són exemples Adobe Creative Cloud, Google Docs o Office365).

4) *Business Process as a Service* (BPaaS): és la capa més nova (i se sustenta damunt de les altres 3), on el model vertical (o horitzontal) d'un procés de negoci es pot oferir sobre una infraestructura *cloud*.

Si bé els avantatges són evidents i molts actors d'aquesta tecnologia la basen principalment en l'abaratiment dels costos de servei, comencen a haver-hi opinions en contra (<http://deepvalue.net/ec2-is-380-more-expensive-than-internal-cluster/>) que consideren que un *cloud* públic potser no és l'opció més adequada per a determinat tipus de serveis/infraestructura. Entre altres aspectes, els negatius poden ser la fiabilitat en la prestació de servei, seguretat i privadesa de les dades/informació en els servidors externs, *lock-in* de dades en la infraestructura sense possibilitat d'extreure-les, estabilitat del proveïdor (com a empresa/negoci) i posició de força al mercat no subjecte a la variabilitat del mercat, colls d'ampolla en les transferències (empresa/proveïdor), rendiment no previsible, temps de resposta a incidents/accidents, problemes derivats de la falta de maduresa de la tecnologia/infraestructura/gestió, acords de serveis (SLA) pensats més per al proveïdor que per a l'usuari, etc. No obstant això, és una tecnologia que té els seus avantatges i que amb l'adequada planificació i presa de decisions, valorant tots els factors que influeixen en el negoci i escapant de conceptes superficials (tots el tenen, tots l'utilitzen, baix cost, expansió il·limitada, etc.), pot ser una elecció adequada per als objectius de l'empresa, el seu negoci i la prestació de serveis en IT que necessita o que forma part de les seves finalitats empresarials.[18]

És molt àmplia la llista de proveïdors de serveis *cloud* en les diferents capes, però d'acord amb la informació de Synergy Research Group el 2014*, els líders al mercat d'infraestructures *cloud* són:

*<https://www.srgresearch.com/articles/amazon-salesforce-and-ibm-lead-cloud-infrastructure-service-segments>

1) *IaaS*: Amazon amb gairebé el 50% i IBM i Rackspace amb valors inferiors al 10% cadascun i Google en valors inferiors.

2) *PaaS*: Salesforce (20%) seguides molt de prop per Amazon, Google i Microsoft.

3) *Private/Híbrid*: IBM (15%), Orange i Fujitsu amb valors propers al 5% cadascun.

4) *Business Process as a Service* (BPaaS): Hi ha diversos operadors, per exemple IBM, Citrix i VMware entre d'altres, però no hi ha dades definides de quota de mercat.

Això significa un canvi apreciable en relació amb les dades del 2012, en què es pot observar una variabilitat de l'oferta molt alta**.

**<https://www.srgresearch.com/articles/amazons-cloud-iaas-and-paas-investments-pay>

Quant a plataformes per a desplegar *clouds* amb llicències GPL, Apache i BSD (o similars), podem comptar entre les més referenciades (i per ordre alfabètic):

- 1) AppScale: és una plataforma que permet als usuaris desenvolupar i executar/emmagatzemar les pròpies aplicacions basades en Google AppEngine i pot funcionar com a servei o en local. Es pot executar sobre AWS EC2, Rackspace, Google Compute Engine, Eucalyptus, Openstack, CloudStack, així com sobre KVM i VirtualBox i suporta Python, Java, Go i plataformes PHP Google AppEngine. <http://www.appscale.com/>
- 2) Cloud Foundry: és una plataforma *open source* PaaS desenvolupada per VMware escrita bàsicament en Ruby and Go. Pot funcionar com a servei i també en local. <http://cloudfoundry.org/>
- 3) Apache CloudStack: dissenyada per a gestionar grans xarxes de màquines virtuals com IaaS. Aquesta plataforma inclou totes les característiques necessàries per al desplegament d'un IaaS: CO (*compute orchestration*), *Network-as-a-Service*, gestió d'usuaris/comptes, API nativa, *resource accounting* i UI millorada (*User Interface*). Suporta els *hypervisors* més comuns, gestió del *cloud* via web o CLI i una API compatible amb AWS EC2/S3 que permeten desenvolupar *clouds* híbrids. <http://cloudstack.apache.org/>
- 4) Eucalyptus: plataforma que permet construir *clouds* privats compatibles amb AWS. Aquest programari permet aprofitar els recursos de còmput, xarxa i emmagatzematge per a oferir autoservei i desplegament de recursos de *cloud* privat. Es caracteritza per la simplicitat en la instal·lació i estabilitat en l'entorn amb una alta eficiència en l'ús dels recursos. <https://www.eucalyptus.com/>
- 5) Nimbus: és una plataforma que una vegada instal·lada sobre un clúster, proporciona IaaS per a la construcció de *clouds* privats o de comunitat i pot ser configurada per a suportar diferents virtualitzacions, sistemes de cues o Amazon EC2. <http://www.nimbusproject.org/>
- 6) OpenNebula: és una plataforma per a gestionar tots els recursos d'un centre de dades i permetre la construcció de IaaS privats, públics i híbrids. Proveeix d'una gran quantitat de serveis, prestacions i adaptacions que han permès que sigui una de les plataformes més difoses en l'actualitat. <http://opennebula.org/>
- 7) OpenQRM: és una plataforma per al desplegament de *clouds* sobre un centre de dades heterogènies. Permet la construcció de *clouds* privats, públics i híbrids amb IaaS. Combina la gestió del la CPU/emmagatzematge/xarxa per a oferir serveis sobre màquines virtualitzades i permet la integració amb recursos remots o altres *clouds*. <http://www.openqrm-enterprise.com/community.html>
- 8) OpenShift: aposta important de la companyia RH i és una plataforma (versió Origin) que permet prestar servei *cloud* en modalitat PasS. OpenShift suporta l'execució de binaris que són aplicacions web tal com s'executen en RHEL, per la qual cosa permet un gran nombre de llenguatges i *frameworks*. <https://www.openshift.com/products/origin>

9) OpenStack és una arquitectura programari que permet el desplegament de *cloud* en la modalitat de IaaS. Es gestiona mitjançant una consola de control via web que permet la provisió i control de tots els subsistemes i l'aprovisionament dels recursos. El projecte iniciat (2010) per Rackspace i NASA és actualment gestionat per OpenStack Foundation i hi ha més de 200 companyies adherides al projecte, entre les quals es troben les grans proveïdores de serveis *cloud* públics i desenvolupadores de SW/HW (ATT, AMD, Canonical, Cisco, Dell, EMC, Ericsson, HP, IBM, Intel, NEC, Oracle, RH, SUSE Linux, VMware, Yahoo, entre d'altres). És una altra de les plataformes més referenciades. <http://www.openstack.org/>

10) PetiteCloud: és una plataforma programari que permet el desplegament de *clouds* privats (petits) i no orientats a dades. Aquesta plataforma pot ser utilitzada sola o en unió amb altres plataformes *cloud* i es caracteritza per la seva estabilitat/fiabilitat i facilitat d'instal·lació. <http://www.petitecloud.org>

11) oVirt: si bé no es pot considerar una plataforma *cloud*, oVirt és una aplicació de gestió d'entorns virtualitzats. Això significa que es pot utilitzar per a gestionar els nodes HW, l'emmagatzematge o la xarxa i desplegar i monitorar les màquines virtuals que s'estan executant al centre de dades. Forma part de RH Enterprise Virtualization i és desenvolupada per aquesta companyia amb llicència Apache. <http://www.ovirt.org/>

2.1. Opennebula

Considerant les opinions de [19],[20], la nostra prova de concepte la farem sobre OpenNebula. Atès que la versió en Debian Wheezy és la 3.4 i el web dels desenvolupadors és el 4.6.2 (<http://opennebula.org/>), hem decidit instal·lar els paquets Debian amb KVM com a *hypervisor* de la nova versió*. Aquesta instal·lació és una prova de concepte (funcional però mínima), però útil per a després fer un desplegament sobre una arquitectura real, on d'una banda executarem els serveis d'OpenNebula i la seva interfície gràfica (anomenada Sunstone) i, d'una altra banda, un hypervisor (*host*) que executarà les màquines virtuals. OpenNebula assumeix dos rols separats: *frontend* i *nodes*. El **frontend** és el que executa els serveis/gestió web i els **nodes** executen la màquina virtual, i si bé en la nostra instal·lació de proves executarem el *frontend* i *nodes* a la mateixa màquina, es recomana executar les màquines virtuals en altres *hosts* que tinguin les extensions de virtualització (en el nostre cas, KVM). Per a verificar si disposem d'aquestes extensions en HW podem executar `grep -E 'svm|vmx' /proc/cpuinfo` i si ens dóna sortida és gairebé segur que el sistema suporta aquestes extensions. Tingueu en compte que això ho haurem de fer sobre un sistema operatiu base (*bare metal*) i no sobre un de ja virtualitzat, és a dir, no podem instal·lar OpenNebula sobre una màquina virtualitzada per VirtualBox, per exemple (hi ha *hypervisors* que permeten aquesta instal·lació com VMware ESXi però VirtualBox, no). Els paquets que conformen la instal·lació són:

*http://docs.opennebula.org/4.6/design_and_installation/quick_starts/qs_ubuntu_kvm.html

- 1) `opennebula-common`: arxius comuns.
- 2) `libopennebula-ruby`: biblioteques de ruby.
- 3) `opennebula-node`: paquet que prepara un node on estarà la VM.
- 4) `opennebula-sunstone`: OpenNebula Sunstone *Web Interface*.
- 5) `opennebula-tools`: *Command line interface*.
- 6) `opennebula-gate`: permet la comunicació entre VMs i OpenNebula.
- 7) `opennebula-flow`: gestiona els serveis i l'“elasticitat”.
- 8) `opennebula`: OpenNebula *daemon*.

Instal·lació del *frontend* (com a *root*): Instal·lar el repositori:

```
wget -q -O- http://downloads.opennebula.org/repo/Debian/repo.key  
| apt-key add -
```

i després els agreguem a la llista

```
echo "deb http://downloads.opennebula.org/repo/Debian/7 stable  
opennebula"> /etc/apt/sources.list.d/opennebula.list
```

Instal·lar els paquets: fem `apt-get update` i després `apt-get install opennebula opennebula-sunstone` (si el node està en un *host* separat, hauran de compartir un directori per NFS, per la qual cosa és necessari també instal·lar el `nfs-kernel-server`).

Serveis: haurem de tenir dos serveis en marxa que són OpenNebula *daemon* (`oned`) i la interfície gràfica (`sunstone`), la qual només està configurada per seguretat per a atendre el *localhost* (si es desitja canviar, cal editar `/etc/one/sunstone-server.conf` i canviar `:host: 127.0.0.1` per `:host: 0.0.0.0` i reiniciar el servidor `/etc/init.d/opennebula-sunstone restart`).

Configurar el NFS (si estem en un únic servidor amb *frontend+Node*, aquesta part no és necessària): s'ha d'afegir a `/etc/exports` del *frontend* `/var/lib/one/*(rw,sync,no_subtree_check,root_squash)` i reiniciar el servei (amb l'ordre `service nfs-kernel-server restart`).

Configurar la clau pública de SSH: OpenNebula necessita accedir per SSH als nodes com l'usuari `oneadmin` i sense `passwd` (des de cada node a un altre, inclòs el *frontend*), per la qual cosa ens canviem com a usuari `oneadmin` (`su - oneadmin`) i executem

```
cp ~/.ssh/id_rsa.pub ~/.ssh/authorized_keys
```

i agreguem el següent text a `~/.ssh/config` (perquè no demani confirmació de `known_hosts`):

```
cat << EOT > ~/.ssh/config
Host *
    StrictHostKeyChecking no
    UserKnownHostsFile /dev/null
EOT
chmod 600 ~/.ssh/config
```

Instal·lació dels nodes: repetim el pas de configurar el repositori i executem `apt-get install opennebula-node nfs-common bridge-utils` (en el nostre cas no és necessari, ja que és la mateixa màquina i només hem d'instalar `opennebula-node` i `bridge-utils`). Hem de verificar que podem accedir com a `oneadmin` a cada node i copiem les claus de `ssh`.

Configuració de la xarxa: (hem de fer un *backup* dels arxius que modificarem prèviament) generalment tindrem `eth0` i la connectarem a un *bridge* (el nom del *bridge* haurà de ser el mateix en tots els nodes) i en `/etc/network/interfaces` agreguem (cal posar les IP que corresponguin):

```
auto lo
iface lo inet loopback
auto br0
iface br0 inet static
    address 192.168.0.10
    network 192.168.0.0
    netmask 255.255.255.0
    broadcast 192.168.0.255
    gateway 192.168.0.1
    bridge_ports eth0
    bridge_fd 9
    bridge_hello 2
    bridge_maxage 12
    bridge_stp off
```

Si tenim servei de DHCP, l'hem de reemplaçar per:

```
auto lo
iface lo inet loopback
auto br0
iface br0 inet dhcp
    bridge_ports eth0
    bridge_fd 9
    bridge_hello 2
    bridge_maxage 12
    bridge_stp off
```

I reiniciem la xarxa `/etc/init.d/networking restart`

Configurem el NFS sobre els nodes (no és necessari en el nostre cas): agreguem a `/etc/fstab`

```
192.168.1.1:/var/lib/one/ /var/lib/one/ nfs soft,intr,rsize=8192,wsiz=8192,noauto
```

on 192.168.1.1 és la IP del *frontend*; després muntem el directori `mount /var/lib/one/`.

Configurem Qemu: l'usuari **oneadmin** ha de poder manejar libvirt com a *root*, per la qual cosa executem el següent.

```
cat << EOT > /etc/libvirt/qemu.conf
user = "oneadmin"
group = "oneadmin"
dynamic_ownership = 0
EOT
```

Reiniciem libvirt amb `service libvirt-bin restart`

Ús bàsic: En el *frontend*, podem connectar-nos a la interfície web en l'URL `http://frontend:9869`. L'usuari és **oneadmin** i el *passwd* està en el seu *HOME* a `/.one/one_auth`, que es genera de manera aleatòria (`/var/lib/one/.one/one_auth`). Per a interactuar amb OpenNebula, s'ha de fer des de l'usuari **oneadmin** i des del *frontend* (per a connectar-se, cal fer simplement `su - oneadmin`).

Agregar un *host*: és el primer que cal fer per després executar VM; es pot fer des de la interfície gràfica o des de CLI fent com **oneadmin** `onehost create localhost -i kvm -v kvm -n dummy` (cal posar el nom del *host* correcte en lloc de *localhost*).

Si hi ha errors, probablement són de `ssh` (hem de verificar que igual que en el cas de **oneadmin**, es pot connectar als altres *hosts* sense *passwd*) i els errors els veurem en `/var/log/one/oned.log`.

El segon pas és agregar la xarxa, una imatge i un *template* abans de llançar una VM:

Xarxa: (es pot fer des de la interfície gràfica també), creem l'arxiu *mynetwork.one* amb el següent contingut:

```
NAME = "private"
TYPE = FIXED
BRIDGE = br0
LEASES = [ IP=192.168.0.100 ]
LEASES = [ IP=192.168.0.101 ]
LEASES = [ IP=192.168.0.102 ]
```

On les IP hauran d'estar lliures a la xarxa en què estem fent el desplegament i executem `onevnet create mynetwork.one`.

En el cas de la imatge, aquesta es pot fer des de CLI però és més simple fer-la des de la interfície web. Seleccionem **Marketplace** i després **ttlinux-kvm** (és una imatge petita de prova) i després li diem **Import**. Podrem verificar que la imatge està en l'apartat corresponent, però l'estat és **LOCK** fins que l'hagi baixat i, quan finalitzi, veurem que canvia a **READY**. Després podrem crear un *template* seleccionant aquesta imatge, la xarxa, i allà podrem posar la

clau pública del `~/.ssh/id_dsa.pub` en l'apartat **Context**, i des d'aquest apartat (**Templates**) podrem dir-li que creï una instància d'aquest *template* (o si no, en **VirtualMachines** crear-ne una utilitzant aquest *template*). Veurem que la VM passa de PENDING – > PROLOG – > RUNNING (si falla en podrem veure en els *logs* la causa) i després ens podrem connectar o bé per `ssh` a la IP de la VM o mitjançant VNC en la interfície web.

Com s'ha pogut comprovar, la instal·lació està prou automatitzada però s'ha de tenir en compte que és una infraestructura complexa i que s'ha de dedicar temps i anàlisi a determinar les causes per les quals es produeixen els errors i solucionar-los. Existeix una gran quantitat de documentació i llocs a Internet, però recomanem començar per les fonts [21] i [22].

En aquest apartat, s'ha vist una prova de concepte funcional però OpenNebula és molt extens i flexible, i permet gran quantitat d'opcions/extensions. Mitjançant OpenNebula C12G Labs (<http://c12g.com/>), es podran trobar i descarregar imatges de màquines virtuals creades per OpenNebula* i llistes per a posar-les en funcionament (són les que es veuen des del Marketplace de la interfície web), però en descarregar-les sobre el nostre servidor les podrem utilitzar sempre que les necessitem i no s'hauran de descarregar cada vegada (no s'han de descomprimir ni s'ha de fer res, s'han d'utilitzar tal com estan). Una de les extensions interessants és OpenNebula Zones (anomenada **ozones**), que ens permet una administració centralitzada de múltiples instàncies d'OpenNebula (zones), gestionant diferents dominis administratius. El mòdul és gestionat per l'*oZones administrator*, que és qui administra els permisos a les diferents zones dels usuaris individuals**. La seva configuració pot trobar-se en el web d'OpenNebula***.

*<http://marketplace.c12g.com/appliance>

**<http://archives.opennebula.org/documentation:arxiv:rel3.0:ozones>
***http://docs.opennebula.org/4.4/advanced_administration/multiple_zone_and_virtual_data_centers/zonesmngt.html

3. DevOps

Tal com hem comentat a l'inici d'aquest apartat, Devops es pot considerar, en paraules dels experts, un moviment tant en l'aspecte professional com en l'aspecte cultural del món TI. Si bé no hi ha totes les respostes encara, un administrador trobarà diferents "comunitats" (i ell mateix formarà part d'alguna), que tindran noves necessitats de desenvolupament i producció de serveis/productes dins del món TI i amb les premisses de "més ràpid", "més eficient", "de major qualitat" i totalment adaptable als "diferents entorns". És a dir, el grup de treball haurà de trencar les barreres entre els departaments d'una empresa i permetre que un producte passi ràpidament des del departament de recerca al de disseny, després al de desenvolupament + producció, després al de test + qualitat i finalment, a vendes, i amb les necessitats d'eines que permetin desplegar totes aquestes activitats en cadascuna de les fases i portar el control de tots pels responsables de cadascun dels àmbits, inclosos els funcionals i els de l'organització / directius. És per això que un administrador necessitarà eines de gestió, control, desplegament, automatització i configuració. Una llista (curta) de les eines que podríem considerar d'acord amb els nostres objectius de llicència GPL-BSD-Apache o similars (és interessant el lloc <http://devs.info/>, ja que permet accedir a la major part d'eines/entorns/documentació per a programadors i una llista més detallada -que inclou SW propietari- es pot trobar a [27]):

- 1) **Linux:** Ubuntu|Debian^v, Fedora^v|CentOS|SL
- 2) **IaaS:** Cloud Foundry, OpenNebula^v, OpenStack
- 3) **Virtualització:** KVM^v, Xen, VirtualBox^v, Vagrant^v
- 4) **Contenidors:** LXC^v, Docker^v
- 5) **Instal·lació SO:** Kickstart i Cobbler (rh), Preseed|Fai^v (deb), Rocks^v
- 6) **Gestió de la configuració:** Puppet^v, Chef^v, CFEngine, SaltStack, Juju, bcfg2, mcollective, fpm (effing)
- 7) **Servidors web i acceleradors:** Apache^v, nginx^v, varnish, squid^v
- 8) **BD:** MySQL^v|MariaDB^v, PostgreSQL^v, OpenLDAP^v, MongoDB, Redis
- 9) **Entorns:** Lamp, Lamr, AppServ, Xampp, Mamp
- 10) **Gestió de versions:** Git^v, Subversion^v, Mercurial^v
- 11) **Monitoratge/supervisió:** Nagios^v, Icinga, Ganglia^v, Cacti^v, Monin^v, MRTG^v, XYmon^v

- 12) Misc: pdsh, pssh, pssh, GNUparallel, nfsroot, Multihost SSH Wrapper, lldpd, Benchmarks^v, Biblioteques^v
- 13) Supervisió: Monit^v, runit, Supervisor, Godrb, BluePill-rb, Upstart, Systemd^v
- 14) Security: OpenVas^v, Tripwire^v, Snort^v
- 15) Desenvolupament i test: Jenkins, Maven, Ant, Gradle, CruiseControl, Hudson
- 16) Desplegament i *workflow*: Capistrano
- 17) Servidors d'aplicacions: JBoss, Tomcat, Jetty, Glassfish,
- 18) Gestió de *logs*: Rsyslog^v, Octopussy^v, Logstash

Excepte les que són molt orientades a desenvolupament d'aplicacions i serveis, en les dues assignatures se n'ha vist (o es veuran dins d'aquest apartat) gran part (marcades amb ^v) i sens dubte, amb els coneixements obtinguts, l'alumne podrà ràpidament desplegar totes aquelles que necessiti i que no s'han tractat en aquests cursos.

A continuació, veurem algunes eines (molt útils en entorns DevOps) més orientades a generar automatitzacions en les instal·lacions o generar entorns aïllats de desenvolupaments/test/execució que permeten de manera simple i fàcil (i sense pèrdues de prestacions/rendiment) disposar d'eines i entorns adequats a les nostres necessitats.

3.1. Linux Containers, LXC

LXC (Linux Containers) és un mètode de virtualització en un nivell del sistema operatiu per a executar múltiples sistemes Linux aïllats (anomenats *contenidors*) sobre un únic *host*. El *kernel* de Linux utilitza *cgroups* per a poder aïllar els recursos (CPU, memòria, E/S, *network*, etc.), la qual cosa no requereix iniciar cap màquina virtual. *Cgroups* també proveeix aïllament dels espais de noms per a aïllar per complet l'aplicació del sistema operatiu, inclosos arbre de processos, xarxa, ID d'usuaris i sistemes d'arxius muntats. Mitjançant una API molt potent i eines simples, permet crear i gestionar contenidors de sistema o aplicacions. LXC utilitza diferents mòduls del *kernel* (*ipc*, *uts*, *mount*, *pid*, *network*, *user*) i d'aplicacions (Apparmor, SELinux profiles, Seccomp policies, Chroots -*pivot_root*- i Control groups -*cgroups*-) per a crear i gestionar els contenidors. Es pot considerar que LXC és a mig camí d'un "potent" *chroot* i una màquina virtual, i ofereix un entorn molt proper a un Linux estàndard però sense necessitat de tenir un kernel separat. Això és més eficient que utilitzar virtualització amb un *hypervisor* (KVM, VirtualBox) i més ràpid de (re)iniciar, sobretot si s'estan fent desenvolupaments i és necessari fer-ho freqüentment, i el seu impacte en el rendiment és molt baix (el contenidor no ocupa recursos) i tots es dediquen als processos que s'estiguin executant. L'únic inconvenient de la utilització de LXC és que només es poden executar

sistemes que suporten el mateix *kernel* que el seu amfitrió, és a dir, no podrem executar un contenidor BSD en un sistema Debian.

La instal·lació es fa mitjançant l'ordre `apt-get install lxc lxcctl` i es poden instal·lar altres paquets addicionals que són opcionals (`bridge-utils` `libvirt-bin` `debootstrap`), no obstant això, si volem que els contenidors tinguin accés a la xarxa, amb IP pròpia és convenient instal·lar `apt-get install bridge-utils`. Per a preparar el *host*, primer hem de muntar el directori `cgroup` afegint a `/etc/fstab` la següent línia `cgroup /sys/fs/cgroup cgroup defaults 0 0` i verificant que podem fer `mount -a` i el sistema *cgroups* apareixerà muntat quan executem l'ordre `mount`. Podeu veure detalls i possibles solucions a errors a [23, 24, 25]. A partir d'aquest moment, podem verificar la instal·lació amb `lxc-checkconfig`, que donarà una sortida similar a:

```
Found kernel config file /boot/config-3.2.0-4-amd64
--- Namespaces ---
Namespaces: enabled
Utsname namespace: enabled
Ipc namespace: enabled
Pid namespace: enabled
User namespace: enabled
Network namespace: enabled
Multiple /dev/pts instances: enabled

--- Control groups ---
Cgroup: enabled
Cgroup clone_children flag: enabled
Cgroup device: enabled
Cgroup sched: enabled
Cgroup cpu account: enabled
Cgroup memory controller: enabled
Cgroup cpuset: enabled

--- Misc ---
Veth pair device: enabled
Macvlan: enabled
Vlan: enabled
File capabilities: enabled
```

Si sorgeixen opcions deshabilitades, s'ha de mirar la causa i solucionar-ho (encara que algunes poden estar-ho). Si bé la distribució inclou un contenidor Debian (`/usr/share/lxc/templates`), és recomanable obtenir-ne un des de l'adreça de <https://github.com/simonvanderveldt/lxc-debian-wheezy-template>, que dona solució a alguns problemes que té l'original quan s'instal·la sobre Debian Wheezy. Per a això, podem fer:

```
wget https://github.com/simonvanderveldt/lxc-debian-wheezy-template/raw/master/lxc-debian-wheezy-robvdhoeven
-O /usr/share/lxc/templates/lxc-debianW
host# chown root:root /usr/share/lxc/templates/lxc-debianW
host# chmod +x /usr/share/lxc/templates/lxc-debianW
```

A continuació, es crea el primer contenidor com a `lxc-create -n mycont -t debianW` (en aquest cas, l'anomenem **mycont** i utilitzem el *template* de

debianW, però també pot ser Debian, que és el que inclou la distribució). Si desitgem configurar la xarxa, primer hem de configurar el *bridge* modificant (en el *host*) */etc/network/interfaces* amb el següent:

```
# The loopback network interface
auto lo br0
iface lo inet loopback

# The primary network interface
iface eth0 inet manual      # la posem en manual per evitar problemes
iface br0 inet static       # inicialitzem el bridge
address 192.168.1.60
netmask 255.255.255.0
gateway 192.168.1.1
bridge_ports eth0
bridge_stp off              # disable Spanning Tree Protocol
    bridge_waitport 0       # no delay before a port becomes available
    bridge_fd 0              # no forwarding delay
```

Fem un `ifdown eth0` i després un `ifup br0` i verifiquem amb `ifconfig` i per exemple `ping google.com` que tenim connectivitat. Després, modifiquem l'arxiu de configuració */var/lib/lxc/mycont/config* per a inserir el *bridge*:

```
lxc.utsname = myvm
lxc.network.type = veth
lxc.network.flags = up
lxc.network.link = br0          #ha d'existir en el host
lxc.network.ipv4 = 192.168.1.100/24 # Ip per al contenidor 0.0.0.0 indica dhcp
lxc.network.hwaddr = 00:1E:1E:1a:00:00
```

Les ordres més útils per a gestionar els contenidors són:

- 1) Per a iniciar l'execució com a *daemon* -en *background*- (el login/passwd per defecte és *root/root*): `lxc-start -n mycont -d`.
- 2) Per a connectar-nos al contenidor: `lxc-console -n mycont "Ctrl+a q"` per a sortir de la consola.
- 3) Per a iniciar el contenidor annexat a la consola: `lxc-start -n mycont` (en *foreground*)
- 4) Per a parar l'execució del contenidor: `lxc-stop -n myvm`
- 5) Per a iniciar els contenidors al *boot* de manera automàtica s'haurà de fer un enllaç de l'arxiu de configuració en el directori */etc/lxc/actel*, per exemple `ln -s /var/lib/lxc/mycont/config /etc/lxc/auto/mycont`
- 6) Per a muntar sistemes d'arxius externs dins del contenidor, agregem a l'arxiu */var/lib/lxc/mycont/config* la línia

```
lxc.mount.entry=/path/in/host/mount_point /var/lib/lxc/mycont/rootfs/mount_moint
none bind 0 0
```

i reiniciem el contenidor.

S'ha d'anar amb compte quan s'inicia el contenidor sense `-d`, ja que no hi ha manera de sortir (en la versió actual, el 'Ctrl+a q' no funciona). Cal iniciar sempre els contenidors en *background* (amb `-d`) tret que necessiti depurar perquè el contenidor no arrenca. Una altra consideració que cal tenir és que l'ordre `lxc-halt` executarà el `telinit` sobre l'arxiu `/run/initctl` si existeix i apagarà el *host*, per la qual cosa cal apagar-lo amb `lxc-stop` i no fer un `shutdown -h now` dins del contenidor, ja que també apagarà el *host*. També existeix un panell gràfic per a gestionar els contenidors via web, <http://lxc-webpanel.github.io/install.html> (en Debian hi ha problemes amb l'apartat de xarxa; alguns d'aquests els soluciona <https://github.com/vaytess/lxc-web-panel/tree/lwp-backup>). Per a això, hem de clonar el lloc

```
git clone https://github.com/vaytess/lxc-web-panel.git
```

i després reemplaçar el directori obtingut per `/srv/lwp` -renombrar aquest abans i fer un `/etc/init.d/lwp restart`). També és important notar que la versió disponible en Debian és la 0.8 i en el web del desenvolupador* és la 1.04, que té molts dels errors corregits i la seva compilació és molt simple (vegeu l'arxiu `INSTALL` dins del paquet) per la qual cosa, si s'hi ha de treballar, es recomana aquesta versió.

*<https://linuxcontainers.org/downloads/>

3.2. Docker

Docker és una plataforma oberta per al desenvolupament, empaquetatge i execució d'aplicacions de manera que es puguin posar en producció o compartir més ràpidament separant aquestes de la infraestructura, de manera que sigui menys costós en recursos (bàsicament espai de disc, CPU i engegada) i que estigui tot preparat per al següent desenvolupador amb tot el que es va posar però res de la part d'infraestructura. Això significarà menys temps per a provar i accelerarà el desplegament escurçant de manera significativa el cicle entre que s'escriu el codi i passa a producció. Docker empra una plataforma (contenidor) de virtualització lleugera amb fluxos de treball i eines que l'ajuden a administrar i implementar les aplicacions i proporcionar una manera d'executar gairebé qualsevol aplicació en forma segura en un contenidor aïllat. Aquest aïllament i seguretat permeten executar molts contenidors de manera simultània en el *host* i, atesa la naturalesa (lleugera) dels contenidors, tot això s'executa sense la càrrega addicional d'un *hypervisor* (que seria l'altra manera de gestionar aquestes necessitats compartint la VM), la qual cosa significa que podem obtenir millors prestacions i utilització dels recursos. És a dir, amb una VM cada aplicació virtualitzada inclou no només l'aplicació, que poden ser desenes de Mbytes -binaris + biblioteques-, sinó també el sistema operatiu *guest*, que poden ser diversos Gbytes; en canvi, en Docker el que es denomina DE (Docker Engine) només comprèn l'aplicació i les seves dependències, que s'executen com un procés aïllat a l'espai d'usuari del SO *host*, compartint el *kernel* amb altres contenidors. Per tant, té el benefici de l'aïllament i l'assignació de recursos de les màquines virtuals, però és molt més portàtil i eficient

transformant-se en l'entorn perfecte per a donar suport al cicle de vida del desenvolupament de programari, test de plataformes, entorns, etc.[33]

L'entorn està format per dos grans components: **Docker** [34] (és la plataforma de virtualització –contenedor) i **Docker Hub** [35] (una plataforma SasS que permet obtenir i publicar/gestionar contenidors ja configurats). En la seva arquitectura, Docker utilitza una estructura client-servidor en què el client (CLI **Docker**) interactua amb el *daemon* Docker, que fa el treball de la construcció, execució i distribució dels contenidors de Docker. Tant el client com el *daemon* es poden executar en el mateix sistema, o es pot connectar un client a un *daemon* de Docker remot. El client Docker i el servei es comuniquen mitjançant *sockets* o una API RESTful.

Per a la instal·lació, no estan disponibles els paquets sobre Debian Wheezy (sí ho estaran sobre Jessie), i hem d'actualitzar el kernel, ja que Docker necessita una versió 3.8 (o superior). Per a això, carregarem el nou kernel 3.14 (juliol del 2014) des de Debian Backports executant:

```
echo "deb http://ftp.debian.org/debian/ wheezy-backports main non-free contrib"
> /etc/apt/sources.list.d/backport.list
apt-get update
apt-get -t wheezy-backports install linux-image-amd64 linux-headers-amd64
reboot
```

Una vegada que hem seleccionat el nou *kernel* durant l'arrencada, podem fer:

```
Instal·lem unes dependències:
apt-get install apt-transport-https
Importem la key d'Ubuntu:
apt-key adv --keyserver hkp://keyserver.ubuntu.com:80 --recv-keys
36A1D7869245C8950F966E92D8576A8BA88D21E9
Agreguem el repositori de Docker:
echo "deb http://get.docker.io/ubuntu docker main" > /etc/apt/sources.list.d/docker.list"
Actualitzem i instal·lem:
apt-get update
apt-get install lxc-docker
Ara podem verificar si la instal·lació funciona executant una imatge d'Ubuntu (local)
dins del seu contenidor:
docker run -i -t ubuntu /bin/bash (Ctrl+D per a sortir del contenidor)
Des de dins, podem executar more lsb-release i ens indicarà:
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=14.04
DISTRIB_CODENAME=trusty
DISTRIB_DESCRIPTION="Ubuntu 14.04 LTS"
```

Però també podem fer `docker run -i -t centos /bin/bash`. Com que la imatge no la té en local, la descarregarà del Docker Hub i l'executarà.

Fent `more /etc/os-release`, ens indicarà que és un CentOS Linux, 7 (Core) i ens donarà informació complementària.

Abans de carregar el primer contenidor, ens pot donar un error com *Cannot start container ... mkdir /sys/fs/devices: operation not permitted*. Això és a causa d'una instal·lació prèvia de LXC i només hem de comentar en l'arxiu `/etc/fstab` la línia `#cgroup /sys/fs/cgroup cgroup defaults 0 0`, reini-

ciar la màquina i executar novament el contenidor. Les ordres per a iniciar-se (a més de les que hem vist) són:

- 1) `docker`: mostra totes les opcions.
- 2) `docker images`: fa una llista de les imatges localment.
- 3) `docker search patró`: busca contenidors/imatges que tinguin aquest patró.
- 4) `docker pull nom`: obté imatges del *hub*. El nom pot ser `<username>/<repository>` per als particulars.
- 5) `docker run name cmd`: executarà el contenidor i a dins, l'ordre indicada per `cmd`.
- 6) `docker ps -l`: permet obtenir l'ID i informació d'una imatge.
- 7) `docker commit ID name`: actualitza la imatge amb què s'ha instal·lat fins a aquest moment (amb 3-4 nombres de l'ID és suficient).
- 8) `docker inspect`: mostra les imatges corrent (similar a `docker ps`).
- 9) `docker push`: salva la imatge en el *hub* (hem de registrar-nos primer) i està disponible per a altres usuaris/*hosts* que la vulguin instal·lar amb tot el que hem configurat dins.
- 10) `docker cp`: copia arxius/directoris des del contenidor al *host*.
- 11) `docker export/import`: exporta/importa el contenidor en un arxiu `tar`.
- 12) `docker history`: mostra la història d'una imatge.
- 13) `docker info`: mostra informació general (imatges, directoris, etc.).
- 14) `docker kill`: per a l'execució d'un contenidor.
- 15) `docker restart`: reinicia un contenidor.
- 16) `docker rm`: elimina un contenidor.
- 17) `docker rmi`: elimina imatges.
- 18) `docker start/stop`: inicia/atura un contenidor.

3.3. Puppet

Puppet és una eina de gestió de configuració i automatització en sistemes TI (licència Apache –abans GPL). El seu funcionament és gestionat mitjançant arxius (anomenats *manifests*) de descripcions dels recursos del sistema i els seus estats, utilitzant un llenguatge declaratiu propi de l'eina. El llenguatge Puppet pot ser aplicat directament al sistema, o compilat en un catàleg i distribuït al sistema de destinació utilitzant un model client-servidor

(mitjançant una API REST), en què un agent interpel·la els proveïdors específics del sistema per a aplicar el recurs especificat en els *manifests*. Aquest llenguatge permet una gran abstracció que habilita els administradors per a descriure la configuració en termes d'alt nivell, com ara usuaris, serveis i paquets sense necessitat d'especificar les ordres del sistema operatiu (`apt`, `dpkg`, etc.).[36, 37, 38] L'entorn Puppet té dues versions: Puppet (*Open Source*) i Puppet Enterprise (producte comercial). Les diferències entre aquestes es poden veure a <http://puppetlabs.com/puppet/enterprise-vs-open-source>. A més s'integren amb aquests entorns altres eines com MCollective (*orchestration framework*), Puppet Dashboard (consola web però abandonada en el seu desenvolupament), PuppetDB (*datawarehouse* per a Puppet), Hiera (eina de cerca de dades de configuració), Facter (eina per a la creació de catàlegs) i Geppetto (IDE *-integrated development environment-* per a Puppet).

3.3.1. Instal·lació

És important començar amb dues màquines que tinguin els seus *hostname* i definició a */etc/hosts* correctament i que siguin accessibles mitjançant la xarxa amb les dades obtingudes del */etc/hosts* (és important que el nom de la màquina -sense domini- d'aquest arxiu coincideixi amb el nom especificat en el *hostname* ja que, si no, hi haurà problemes quan es generin els certificats).

```
Obtenim els repositoris i instal·lem puppetmaster en el servidor:
wget http://apt.puppetlabs.com/puppetlabs-release-wheezy.deb
dpkg -i puppetlabs-release-wheezy.deb
apt-get update
apt-get install puppetmaster
```

```
Puppetlabs recomana executar la següent ordre abans de córrer puppetmaster:
puppet resource service puppetmaster ensure=running enable=true
service puppetmaster restart
```

Sobre el client, s'ha d'instal·lar els repositoris (tres primeres instruccions) i instal·lar puppet (si el client és Ubuntu, no és necessari instal·lar el repositori):

```
apt-get install puppet
```

Sobre el client, s'ha de modificar */etc/puppet/puppet.conf* per a agregar en la secció [main] el servidor (*sysdw.nteum.org* en el nostre cas) i reiniciar:

```
[main]
server=sysdw.nteum.org
/etc/init.d/puppet restart
```

Després, s'ha d'executar sobre el client:

```
puppet agent --waitforcert 60 --test
```

Aquesta ordre enviarà una petició de signatura del certificat al servidor i esperarà que aquest l'hi signi, per la qual cosa, ràpidament sobre el servidor s'ha de fer:

```
puppetca --list
```

Ens respondrà amb alguna cosa com "pucli.nteum.org" (...) i executem:

```
puppetca --sign pucli.nteum.org
```

Veurem informació per part del servidor i també per part del client i l'execució dels catàlegs que hi hagi pendants.

D'aquesta manera, tenim instal·lat el client i el servidor i ara haurem de fer el nostre primer 'manifest'. Per a això, organitzarem l'estructura d'acord amb les recomanacions de PuppetLabs fent un arbre que tindrà la següent estructura a partir de */etc/puppet*:

```

+-- auth.conf
+-- autosign.conf
+-- environments
|   +-- common
|   +-- development
|   |   +-- modules
|   +-- production
|       +-- modules
|       +-- ntp
+-- etckeeper-commit-post
+-- etckeeper-commit-pre
+-- files
|   +-- shadow.sh
+-- filesserver.conf
+-- manifests
|   +-- nodes
|   |   +-- client1.pp
|   |   +-- server.pp
|   +-- site.pp
+-- modules
|   +-- accounts
|   |   +-- manifests
|   |   +-- init.pp
|   |   +-- system.pp
|   +-- elinks
|   |   +-- manifests
|   |   +-- init.pp
|   +-- nmap
|       +-- manifests
|       +-- init.pp
+-- node.rb
+-- puppet.conf
+-- rack
+-- templates

```

Començarem per l'arxiu */etc/puppet/manifests/site.pp*, que com a contingut tindrà `import 'nodes/*.pp'`. En el directori */etc/puppet/manifests/nodes* tindrem dos arxius (*server.pp* i *client1.pp*) amb el següent contingut:

```

Arxiu server.pp:
  node 'sysdw.nteum.org' {
  }

Arxiu client1.pp:
  node 'pucli.nteum.org' {
    include nmap
    include elinks
  }

```

On definim dos serveis que cal instal·lar en el client *pucli.nteum.org* (*nmap* i *elinks*). Després, definim els directoris */etc/puppet/modules/elinks/manifests*, i */etc/puppet/modules/nmap/manifests* on hi haurà un arxiu *init.pp* amb la tasca que s'ha de fer:

```

Arxiu /etc/puppet/modules/elinks/manifests/init.pp:
class elinks {
  case $operatingsystem {
    centos, redhat: {
      package { [ " elinks":
      ensure => installed,}}
    debian, ubuntu: {
      package { [ " elinks":
      ensure => installed,}}
  }
}

```

```

    }
}
Arxiu /etc/puppet/modules/nmap/manifests/init.pp:
class nmap {
  case $operatingsystem {
    centos, redhat: {
      package { "nmap":
        ensure => installed,}}
    debian, ubuntu: {
      package { "nmap":
        ensure => installed,}}
  }
}

```

En aquests, podem veure la selecció del SO i després les indicacions del paquet que cal instal·lar. Tot això es podria haver posat en l'arxiu inicial (*site.pp*), però es recomana fer-ho així per a millorar la visibilitat/estructura i que puguin aprofitar-se les dades per a diferents instal·lacions. Sobre el servidor, hauríem d'executar `puppet apply -v /etc/puppet/manifests/site.pp`, i sobre el client, per a actualitzar el catàleg (i instal·lar les aplicacions) `puppet agent -v -test` (en cas de no fer-ho, el client s'actualitzarà al cap de 30 minuts per defecte). Es podrà verificar sobre el client que els paquets s'han instal·lat i fins i tot es poden desinstal·lar des de la línia d'ordres i executar l'agent que els tornarà a instal·lar.[37]

Com a següent acció, procurarem crear un usuari i posar-li un *password*. Comencem creant un mòdul */etc/puppet/modules/accounts/manifests* i, dins d'aquest, dos arxius anomenats *init.pp* i *system.pp* amb el següent contingut:

```

Arxiu /etc/puppet/modules/accounts/manifests/system.pp:
define accounts::system ($comment,$password) {
  user { $title:
    ensure => 'present',
    shell => '/bin/bash',
    managehome => true,}
}
Arxiu /etc/puppet/modules/accounts/manifests/init.pp:
class accounts {
  file { ['/etc/puppet/templates/shadow.sh':
    ensure => file,
    recurse => true,
    mode => "0777",
    source => "puppet:///files/shadow.sh",}
  @accounts::system { 'demo':
    comment => 'demo users',
    password => '*',}
  exec { "demo":
    command => 'echo "demo:123456" | chpasswd',
    provider => 'shell',
    onlyif => " /etc/puppet/templates/shadow.sh demo",}
}

```

En l'arxiu *system*, hem definit el tipus *accounts::system* per a assegurar que cada usuari té directori HOME i *shell* (en lloc dels valors predeterminats de l'ordre *useradd*), contrasenya i el camp Gecos. Després, en el *init.pp* indiquem el nom de l'usuari que cal crear i el camp Geco. Per a crear aquest usuari sense *passwd*, hem de modificar *client1.pp* perquè quedi:

```
Arxiu client1.pp:
  node 'pucli.nteum.org' {
    #include nmap
    #include elinks
  include accounts
    realize (Accounts::System['demo'])
  }
```

Com es pot observar, hem comentat el que no volem que s'executi, ja que el `nmap` i l'`elinks` es van instal·lar en l'execució anterior. Aquí indiquem que s'inclogui el mòdul *accounts* i que es faci l'acció. Amb això només creariem usuaris, però no els modificariem el *password*. Hi ha diferents maneres d'inicialitzar el *password*, ja que només s'ha d'executar una vegada i quan no estigui inicialitzat i aquí en seguirem una, vista a [37]. Per a això, utilitzarem un *script* que haurem d'enviar des del servidor al client, per la qual cosa hem de fer:

```
Arxiu /etc/puppet/filesserver.conf modificar:
  [files]
    path /etc/puppet/files
    allow *
Arxiu /etc/puppet/auth.conf incloure:
  path /files
  auth *
Arxiu /etc/puppet/puppet.conf en la secció [main] incloure:
  pluginsync = true
Tornar a arrencar el servidor: /etc/init.d/puppetmaster restart
```

Ara crearem un *script* `/etc/puppet/files/shadow.sh` que ens permetrà saber si hem de modificar el *passwd* del `/etc/shadow` o no:

```
#!/bin/bash
rc=` /bin/grep $1 /etc/shadow | awk -F":" '{ $2 == " ! " }' | wc -l `
if [ $rc -eq 0 ]
then
  exit 1
else
  exit 0
fi
```

I ja tenim les modificacions en l'arxiu `/etc/puppet/modules/accounts/init.pp` on li diem quin arxiu cal transferir i on (*file*) i després l'exec, que permet canviar el *passwd* si l'*script* ens retorna un 0 o un 1. Després haurem d'executar novament sobre el servidor `puppet apply -v /etc/puppet/manifests/site.pp` i sobre el client per a actualitzar el catàleg `puppet agent -v -test` i verificar si l'usuari s'ha creat i podem accedir-hi. Com a complement a Puppet, hi ha el Puppet Dashboard, però aquest programari està sense manteniment, per la qual cosa no es recomana instal·lar-lo (tret que sigui estrictament necessari –indicacions en <https://wiki.debian.org/puppetdashboard>). En el seu lloc es pot instal·lar **Foreman** [39, 40], que és una aplicació que s'integra molt bé amb Puppet i permet veure en una consola tota la informació i seguiment, així com fer el desplegament i l'automatització de manera gràfica d'una instal·lació. La instal·lació és molt simple [40, 41]:

```
Inicialitzem el repositori i instal·lem el paquet foreman-installer:
echo "deb http://deb.theforeman.org/ wheezy stable" > \
/etc/apt/sources.list.d/foreman.list
wget -q http://deb.theforeman.org/foreman.asc -O- | apt-key add -
apt-get update
apt-get install -i foreman-installer
```

Executem l'instal·lador amb -i (interactive) per a verificar els setting:

```
foreman-installer -i
```

Contestem (y) a la pregunta i seleccionem les opcions:
1. Configure foreman, 2. Configure foreman_proxy, 3. Configure puppet.
Veurem que el procés pot acabar amb uns errors (Apache), però no hem de preocupar-nos
i hem de reiniciar el servei amb service apache2 restart.

Ens connectem a l'URL <https://sysdw.nteum.org/>, acceptem el certificat i entrem
amb usuari admin i passwd changeme.
Sobre Infrastructure > Smart Proxy fem clic en New Proxy i seleccionem el nom
i l'URL: <https://sysdw.nteum.org:8443>.

El següent és fer que s'executi el puppet sobre el servidor (si no està posat
en marxa, s'ha d'engegar service puppet start i pot ser necessari modificar
l'arxiu /etc/default/puppet i executem puppet agent -t).

Sobre Foreman, veurem que el DashBoard canvia i actualitza els valors per a la gestió
integrada amb puppet.

3.4. Chef

Chef és una altra de la grans eines de configuració amb funcionalitats similars a Puppet (hi ha un bon article en què s'efectua una comparació sobre el dilema de Puppet o Chef [42]). Aquesta eina utilitza un llenguatge (basat en Ruby) DSL (*domain-specific language*) per a escriure les “receptes” de configuració que seran utilitzades per a configurar i administrar els servidors de la companyia/institució i que a més pot integrar-se amb diferents plataformes (com Rackspace, Amazon EC2, Google i Microsoft Azure) per a, automàticament, gestionar la provisió de recursos de noves màquines. L'administrador comença escrivint les receptes que descriuen com maneja Chef les aplicacions del servidor (com ara Apache, MySQL, o Hadoop) i com seran configurades, i indica quins paquets hauran de ser instal·lats, els serveis que hauran d'executar-se i els arxius que s'hauran de modificar, i a més informará de tot això en el servidor perquè l'administrador tingui el control del desplegament. Chef es pot executar en mode client/servidor o en mode *standalone* (anomenat Chef-solo). En el mode C/S, el client envia diversos atributs al servidor i el servidor utilitza la plataforma `solr` per a indexar aquests i proveir l'API corresponent, i utilitza aquests atributs per a configurar el node. Chef-solo permet utilitzar les receptes en nodes que no tinguin accés al servidor i només necessita “la seva” recepta (i les seves dependències), que ha d'estar en el disc físic del node (Chef-solo és una versió amb funcionalitat limitada de Chef-client). Chef, juntament amb Puppet, CFEngine i Bcfg2, és una de les eines més utilitzades per a aquest tipus de funcionalitat i que ja forma part de les mitjanes-grans instal·lacions actuals de GNU/Linux (és interessant veure en els detalls del desplegament de la infraestructura Wikipedia que, excepte *passwords* i certificats, tot està documentat a <http://blog.wikimedia.org/2011/09/19/ever-wondered-how-the-wikimedia-servers-are-configured/>).

La instal·lació bàsica de Chef-server i Chef-client pot ser una mica més complicada, però comencem igualment amb màquines que tinguin */etc/hosts* i *hostname* ben configurades. En aquest cas, suggerim treballar amb màquines virtuals Ubuntu (14.04 si pot ser), ja que hi ha dependències de la biblioteca Libc (necessita la versió 2.15) i Ruby (1.9.1) que en Debian Wheezy generen problemes (fins i tot utilitzant el repositori experimental). El primer pas és baixar-nos els dos paquets de <http://www.getchef.com/chef/install/> i executar en cadascun `dpkg -i paquet-corresponent.deb` (és a dir, el servidor a la màquina servidora i el client a la màquina client). Després de cert temps en el servidor, podem executar `chef-server-ctl reconfigure`, que reconfigurarà tota l'eina i crearà els certificats. Quan acabi, podrem connectar-nos a l'URL <https://sysdw.nteum.org/> amb usuari i password `admin/psswOrd1`.

Sobre la interfície gràfica, cal anar a *Client > Create* i crear un client amb el nom desitjat (pucli en el nostre cas), marcar la casella Admin i fer *Create Client*. Sobre la següent pantalla es generaran les claus privades i pública per a aquest client i haurèm de copiar i salvar la clau privada en un arxiu (per exemple */root/pucli.pem*). Sobre el client, haurèm de crear un directori */root/.chef* i des del servidor li copièm la clau privada `scp /root/pucli.pem pucli:/root/.chef/pucli.pem`. Sobre el client fem `knife configuri` i les dades hauran de quedar com:

```
log_level           :info
log_location        STDOUT
node_name           'pucli'
client_key           '/root/.chef/pucli.pem'
validation_client_name 'chef-validator'
validation_key       '/etc/chef/validation.pem'
chef_server_url      'https://sysdw.nteum.org'
cache_type           'BasicFile'
cache_options( :path => '/root/.chef/checksums' )
cookbook_path [ '/root/chef-repo/cookbooks' ]
```

Després podrem executar `knife client list` i ens haurà de mostrar els clients, una cosa com:

```
chef-validator
chef-webui
pucli
```

A partir d'aquest punt, estem en condicions de crear la nostra primera recepta (*cookbook*) però donada la complexitat de treballar amb aquest paquet i els coneixements necessaris, recomanem començar treballant en el client com a Chef-solo per a automatitzar l'execució de tasques i passar després a descriure les tasques del servidor i executar-les remotament. Per a això, recomanem seguir la guia a <http://gettingstartedwithchef.com/>. [44, 43, 45]. Si bé s'han de fer alguns canvis quant als *cookbook* que s'han d'instal·lar, la seqüència és correcta i permet aprendre a treballar i repetir els resultats en tants servidors com sigui necessari (es recomana fer el procediment en un i després sobre una altra

màquina sense cap paquet, instal·lar el Chef-solo, copiar el repositori del primer i executar el Chef-solo per a tenir un altre servidor instal·lat exactament igual que el primer).

3.5. Vagrant

Aquesta eina és útil en entorns DevOps i té un paper diferent del de Docker però orientat cap als mateixos objectius: proporcionar entorns fàcils de configurar, reproduïbles i portàtils amb un únic flux de treball que ajudarà a maximitzar la productivitat i la flexibilitat en el desenvolupament d'aplicacions/serveis. Pot proveir de màquines de diferents proveïdors (VirtualBox, VMware, AWS, o altres) utilitzant *scripts*, Chef o Puppet per a instal·lar i configurar automàticament el programari de la VM. Per als desenvolupadors, Vagrant aïllarà les dependències i configuracions dins d'un únic entorn disponible, consistent, sense sacrificar cap de les eines que el desenvolupador utilitza habitualment i tenint en compte un arxiu anomenat *Vagrantfile*. La resta de desenvolupadors tindrà el mateix entorn encara que treballin des d'uns altres SO o entorns, de manera que s'aconseguirà que tots els membres d'un equip estiguin executant codi en el mateix entorn i amb les mateixes dependències. A més, en l'àmbit TI permet tenir entorns d'un sol ús amb un flux de treball coherent per a desenvolupar i provar *scripts* d'administració de la infraestructura, ja que ràpidament es poden fer proves de *scripts*, *cookbooks* de Chef, mòduls de Puppets i altres que utilitzen la virtualització local, com VirtualBox o VMware. Després, amb la mateixa configuració, pot provar aquests *scripts* en el *cloud* (per exemple, AWS o Rackspace) amb el mateix flux de treball.

En primer lloc, s'ha d'indicar que haurem de treballar sobre un GNU/Linux base (el que es defineix com a *bare metal*, és a dir, sense virtualitzar, ja que necessitem les extensions de HW visibles i si hem virtualitzat amb VirtualBox, per exemple, això no és possible). És recomanable instal·lar la versió de VirtualBox (pot ser la del repositori Debian), verificar que tot funciona i després descarregar l'última versió de Vagrant des de <http://www.vagrantup.com/downloads.html> i instal·lar-la amb `dpkg -i vagrant_x.y.z_x86_64.deb` (on x.y.z serà la versió que hem descarregat).

A partir d'aquest punt, és molt simple executant l'ordre [46] `vagrant init hashicorp/precisi32`, que inicialitzarà/generarà un arxiu anomenat **Vagrantfile** amb les definicions de la VM, i quan executem **vagrant up** descarregarà del repositori *cloud* de Vagrant una imatge d'Ubuntu 12.04-32b i l'engegarà. Per a accedir-hi, simplement hem de fer `vagrant ssh`, i si volem rebutjar-la, `vagrant destroy`. En el *cloud* de Vagrant [47], podem accedir a diferents imatges preconfigurades* que podrem carregar simplement fent `vagrant box add nom`, per exemple `vagrant box add chef/centos-6.5`. Quan s'executi l'ordre `up` veurem que ens dóna una sèrie de missatges, entre els quals ens indicarà el port per a connectar-se a aquesta màquina virtual directament (per exemple, `ssh vagrant@localhost -p 2222` i amb *passwd*

*<https://vagrantcloud.com/discover/featured>

vagrant). Per a utilitzar una VM com a base, podem modificar l'arxiu *Vagrantfile* i canviar-ne el contingut:

```
Vagrant.configure("2") do |config|
  config.vm.box = "hashicorp/precise32"
end
```

Si desitgem treballar amb dues màquines de manera simultània, haurem de modificar l'arxiu *Vagrantfile* amb el següent:

```
Vagrant.configure("2") do |config|
  config.vm.define "centos" do |centos|
    centos.vm.box = "chef/centos-6.5"
  end
  config.vm.define "ubu" do |ubu|
    ubu.vm.box = "hashicorp/precise32"
  end
end
```

Podrem veure que assigna un port SSH a cadascuna per a evitar col·lisions quan fem el `vagrant up`. Des de la VM podríem instal·lar el programari com es fa habitualment, però per a evitar que cada persona faci el mateix hi ha una forma d'aprovisionament que s'executarà quan es faci el `up` de la VM. Per a això, escrivim un *script* `init.sh` amb el següent contingut:

```
#!/usr/bin/env bash
apt-get update
apt-get install -y apache2
rm -rf /var/www
ln -fs /tmp /var/www
```

En aquest *script* (per a no tenir problemes de permisos per a aquesta prova) hem apuntant el directori `/var/www` a `/tmp`. A continuació, modifiquem el *Vagrantfile* (només hem deixat una VM per accelerar els processos de càrrega):

```
Vagrant.configure("2") do |config|
  config.vm.define "ubu" do |ubu|
    ubu.vm.box = "hashicorp/precise32"
    ubu.vm.provision :shell, path: "init.sh"
  end
end
```

Després, haurem de fer `vagrant reload -provision`. Veurem com s'aprovisiona i carrega Apache i després, si entrem a la màquina i creem un fitxer `/tmp/index.html` i fem `wget 127.0.0.1` (o la IP interna de la màquina), veurem que accedim a l'arxiu i el baixem (igualment, si fem `ps -edaf | grep apache2` veurem que està funcionant). Finalment, i per a veure-la des del *host*, hem de redirigir el port 80 per exemple al 4444. Per a això, hem d'agregar en el mateix arxiu (sota `ubu.vm.provision`) la línia:

```
config.vm.network :forwarded_port, host: 4444, guest: 80.
```

Tornem a fer `vagrant reload -provision` i des del *host* ens connectem a l'URL: `127.0.0.1:4444` i veurem el contingut de l'*index.html*.

En aquestes proves de concepte, hem mostrat alguns aspectes interessants però és una eina molt potent que s'ha d'analitzar amb cura per a configurar les opcions necessàries per al nostre entorn, com per exemple aspectes del *Vagrant Share* o qüestions avançades del *Provisioning* que aquí només hem tractat superficialment.[46]

Activitats

1. Instal·leu i configureu OpenMPI sobre un node; compileu i executeu el programa `cpi.c` i observeu el seu comportament.
2. Instal·leu i configureu OpenMP; compileu i executeu el programa de multiplicació de matrius (<https://computing.llnl.gov/tutorials/openMP/exercise.html>) en 2 *cores* i obteniu proves de la millora en l'execució.
3. Utilitzant dos nodes (pot ser amb VirtualBox), Cacti i XYmon i monitoreu el seu ús.
4. Utilitzant Rocks i VirtualBox, instal·leu dues màquines per a simular un clúster.
5. Instal·leu Docker i creeu 4 entorns diferents.
6. Instal·leu Puppet i configureu un màquina client.
7. Ídem que en el punt anterior amb Chef.
8. Amb Vagrant creeu dues VM, una amb Centos i una altra amb Ubuntu i proveïeu la primera amb Apache i la segona amb MySQL. S'ha de poder accedir a les màquines des del *host* i s'han de poder comunicar entre elles.

Bibliografia

- [1] *Beowulf cluster*.
<http://en.wikipedia.org/wiki/Beowulf_cluster>
- [2] **S. Pereira**. *Building a simple Beowulf cluster with Ubuntu*.
<http://byobu.info/article/Building_a_simple_Beowulf_cluster_with_Ubuntu/>
- [3] **Radajewski, J.; Eadline, D.** *Beowulf: Installation and Administration*. TLDP.
<<http://www2.ic.uff.br/~vefr/research/clcomp/Beowulf-Installation-and-Administration-HOWTO.html>>
- [4] **Swendson, K.** *Beowulf HOWTO* (tldp).
<<http://www.tldp.org/HOWTO/Beowulf-HOWTO/>>
- [5] **Barney, B.** *OpenMP*. Lawrence Livermore National Laboratory.
<<https://computing.llnl.gov/tutorials/openMP/>>
- [6] *OpenMP Exercise*.
<<https://computing.llnl.gov/tutorials/openMP/exercise.html>>
- [7] *MPI 3*.
<<http://www.mpi-forum.org/docs/mpi-3.0/mpi30-report.pdf>>
- [8] *MPI Examples*.
<<http://www.mcs.anl.gov/research/projects/mpi/usingmpi/examples/main.htm>>
(Download Section)
- [9] *Mpich1 Project*.
<<http://www.mcs.anl.gov/research/projects/mpi/>>
- [10] *MPICH, High-Performance Portable MPI*.
<<http://www.mpich.org/>>
- [11] *LAM/MPI*.
<<http://www.lam-mpi.org/>>
- [12] *OpenMPI*.
<<http://www.open-mpi.org/>>
- [13] *FAQ OpenMPI*.
<<http://www.open-mpi.org/faq/>>
- [14] **Woodman, L.** *Setting up a Beowulf cluster Using Open MPI on Linux*.
<<http://techtinkering.com/articles/?id=32>>
- [15] *Rocks Cluster. Base Users Guide*.
<<http://central6.rocksclusters.org/roll-documentation/base/6.1.1/>>
- [16] *Cacti*.
<<http://www.cacti.net/>>
- [17] *Xymon*.
<<http://xymon.sourceforge.net/>>
- [18] *Cloud Computing. Retos y Oportunidades*.
<http://www.ontsi.red.es/ontsi/sites/default/files/2-_resumen_ejecutivo_cloud_computing_vf.pdf>
- [19] *Eucalyptus, CloudStack, OpenStack and OpenNebula: A Tale of Two Cloud Models*.
<<http://opennebula.org/eucalyptus-cloudstack-openstack-and-opennebula-a-tale-of-two-cloud-models/>>
- [20] *OpenNebula vs. OpenStack: User Needs vs. Vendor Driven*.
<<http://opennebula.org/opennebula-vs-openstack-user-needs-vs-vendor-driven/>>
- [21] *OpenNebula Documentation*.
<<http://opennebula.org/documentation/>>
- [22] *Quickstart: OpenNebula on Ubuntu 14.04 and KVM*.
<http://docs.opennebula.org/4.6/design_and_installation/quick_starts/qs_ubuntu_kvm.html>
- [23] *LXC*.
<<https://wiki.debian.org/LXC>>

- [24] *LXC en Ubuntu.*
<<https://help.ubuntu.com/lts/serverguide/lxc.html>>
- [25] *LXC: Step-by-Step Guide.*
<<https://www.stgraber.org/2013/12/20/lxc-1-0-blog-post-series/>>
- [26] **S. Graber.** *LXC 1.0.*
<<https://wiki.debian.org/BridgeNetworkConnections>>
- [27] *A Short List of DevOps Tools.*
<<http://newrelic.com/devops/toolset>>
- [28] *Preseeding d-i en Debian.*
<<https://wiki.debian.org/DebianInstaller/Preseed>>
- [29] *FAI (Fully Automatic Installation) for Debian GNU/Linux.*
<<https://wiki.debian.org/FAI>>
- [30] *FAI (Fully Automatic Installation) project and documentation.*
<<http://fai-project.org/>>
- [31] *El libro del administrador de Debian. Instalación automatizada.*
<<http://debian-handbook.info/browse/es-ES/stable/sect.automated-installation.html>>
- [32] *Octopussy. Open Source Log Management Solution.*
<<http://8pussy.org/>>
- [33] *Understanding Docker.*
<<https://docs.docker.com/introduction/understanding-docker/>>
- [34] *Docker Docs.*
<<https://docs.docker.com/>>
- [35] *Docker HUB.*
<<https://registry.hub.docker.com/>>
- [36] *Puppet Labs Documentation.*
<<http://docs.puppetlabs.com/>>
- [37] *Puppet - Configuration Management Tool.*
<<http://puppet-cmt.blogspot.com.es/>>
- [38] *Learning Puppet.*
<<http://docs.puppetlabs.com/learning/ral.html>>
- [39] *Foreman - A complete lifecycle management tool.*
<<http://theforeman.org/>>
- [40] *Foreman - Quick Start Guide.*
<http://theforeman.org/manuals/1.5/quickstart_guide.html>
- [41] *How to Install The Foreman and a Puppet Master on Debian Wheezy.*
<<http://midactstech.blogspot.com.es/2014/02/PuppetMasterForeman-Wheezy.html>>
- [42] *Puppet or Chef: The configuration management dilemma.*
<<http://www.infoworld.com/d/data-center/puppet-or-chef-the-configuration-management-dilemma-215279?page=0,0>>
- [43] *Download Chef: Server and Client.*
<<http://www.getchef.com/chef/install/>>
- [44] **Gale, A.** *Getting started with Chef.*
<<http://gettingstartedwithchef.com/>>
- [45] *Chef Cookbooks.*
<<https://supermarket.getchef.com/cookbooks-directory>>
- [46] *Vagrant: Getting Started.*
<<http://docs.vagrantup.com/v2/getting-started/index.html> >
- [47] *Vagrant Cloud.*
<<https://vagrantcloud.com/>>
- [48] *KVM.*
<<https://wiki.debian.org/es/KVM>>
- [49] *VirtualBox.*
<<https://wiki.debian.org/VirtualBox>>

-
- [50] *Guía rápida de instalación de Xen.*
<<http://wiki.debian.org/Xen>>
- [51] *The Xen hypervisor.*
<<http://www.xen.org/>>
- [52] *Qemu.*
<http://wiki.qemu.org/Main_Page>
- [53] *QEMU. Guía rápida de instalación e integración con KVM y KQemu.*
<<http://wiki.debian.org/QEMU>>

