SL

EL SOFTWARE LIBRE
HECHO REVISTA

www.revista-sl.org

Diciembre 2006-Enero 2007

Revista SL entrevista a Matt Dillon

PC-BSD Informática personal

Instala OpenBSD paso a paso

Sol, playa y software libre.GULEV

RevistaSL

El software libre hecho revista

~# Editorial

Hablar de los BSD's es hablar de un gran trabajo colaborativo e historia.

Basado en el Unix “puro” desarrollado en los laboratorios Bell de AT&T, lo cual le da una gran autoridad en el mundo “Unix”. Berkeley Software Distribution (Distribución de Software Berkeley) comenzó como la compilación de los agregados por Bill Joy, estudiante graduado en Berkeley, al sistema operativo Unix, que en sus inicios eran aplicaciones a nivel usuario.

Al BSD le debemos: el manejo de memoria virtual paginado por demanda, el control de trabajos, el Fast FileSystem, el protocolo TCP/IP (casi todas las implementaciones de TCP derivan de la de 4.4BSD-Lite) y el editor de texto vi. Por lo que cualquiera que utiliza un sistema Operativo, “Unix” o no-Unix, está usando algo de BSD.

Una demanda de AT&T en 1992 hacia los BSD de la época, 386BSD y BSD/386 (ahora BSD/OS), provocó que los BSD no se desarrollaran y difundieran como lo fue con GNU/Linux, pues duró aproximadamente 2 años el litigio.

Del 386BSD, se derivaron NetBSD y FreeBSD. Es difícil enumerar los objetivos de cada proyecto puesto que las diferencias son muy subjetivas.

En general; FreeBSD tiene como meta ofrecer alto rendimiento y facilidad de uso al usuario final y es uno de los favoritos entre proveedores de contenidos web. NetBSD tiene como meta la Portabilidad: No en vano su lema es “of course it runs NetBSD” (que podría traducirse como “claro que funciona con NetBSD”). OpenBSD tiene como meta la seguridad y la integridad del código: combina del concepto de código abierto y una revisión rigurosa del código que dan como fruto un sistema muy correcto, elegido por instituciones preocupadas por la seguridad.

Existen dos sistemas operativos BSD más que no son de código abierto, BSD/OS y el MacOS X de Apple; BSD/OS es el derivado más antíguo de 4.4BSD. No es código abierto pero es posible conseguir licencias de su código fuente a un precio relativamente bajo. Se parece a FreeBSD en muchos aspectos. Mac OS X es la última versión del sistema operativo para la gama Macintosh de Apple Computer Inc. El núcleo BSD Unix de éste sistema operativo, Darwin, está libremente disponible como sistema operativo de fuente abierto totalmente funcional para arquitecturas x86 y PPC. Varios desarrolladores de Darwin son también “committers” de FreeBSD y viceversa.

Tal vez, sino fuera por una demanda, en estos momentos estaríamos usando un sistema operativo BSD o ¿GNU/BSD?.

www.revista-sl.org

buzon@revista-sl.org

contenido

¿Sabias qué?

curiosidades del software libre

Julio César Sosa **julio.cesar@revista-sl.org**

¿Sabias que el protocolo TCP/IP sobre el cual se sustenta la Internet que conocemos hoy en día fue desarrollado bajo y con tecnología BSD?

Pues bien el protocolo TCP/IP fue desarrollado en 1972 pero fue lanzado al mundo por primera vez en el año de 1983 como una implementación más de la versión 4.2 de BSD la cual fue utilizada como plataforma de desarrollo por parte de la "Agencia de Investigación de Proyectos Avanzados de Defensa de Estados Unidos" o simplemente DARPA (apuesto que les recordó sus primeras clases de historia de la informática y el Internet n_n) cabe indicar que la gran mayoría de las implementaciones del protocolo TCP/IP en gran número de los sistemas operativos conocidos están basadas en el original desarrollado hace ya unos 24 años.

En Internet podemos usar y en cierto sentido disfrutar de la aplicación de muchos otros protocolos que nos brindan la gran mayoría de los servicios que podemos encontrar en la Internet como pueden ser el protocolo DNS que mirándolo desde un plano de usuario final le hace la vida mas fácil a las personas en la red, el FTP por el cual muchos de nosotros podemos enviar y recibir archivos y/o almacenarlos en algún lugar remoto, el HTTP gracias al cual podemos apreciar esas hermosas y otras no tanto páginas Web que a todos nos informan o divierten, el protocolo de IRC por el cual podemos conversar, conocer personas y compartir información. No alcanzarían todas estas páginas para enumerar aun que sea de una manera burda y sumamente coloquial todo los servicios y protocolos que se sustentan gracias al conjunto de protocolos llamado TCP/IP.

Quien lo diría, siendo honestos existe gente a quién le es desconocido BSD y las grandes aportaciones que dicho sistema a realizado al mundo de la Informática y las computadoras, existen una gran cantidad de sistemas operativos derivados del original BSD desarrollado en un principio por la Universidad de Berkeley entre esos sistemas podemos enumerar a NetBSD, FreeBSD y OpenBSD en este caso no se podrían considerar Distribuciones o Meta Distribuciones como bien ocurre con su primo GNU/Linux ya que a pesar de que conservan muchas características del BSD, cada uno dispone de un Kernel diferente. Por desgracia yo no soy un diablito mas bien visto de Frac con pantuflas amarillas. >-< así que si cometí o dije alguna barbaridad pido disculpas a todos esos diablitos que abundan en la red, BSD es un sistema operativo maduro, que puede ser implementado en un sinfín de Arquitecturas computacionales, puede ser escalable y adaptado a un sinnúmero de circunstancias además de que gosa de un inmejorable prestigio como un sistema con el cual se puede crear una infraestructura de seguridad muy confiable, solo basta preguntarles a los amigos de OpenBSD y ese muro de titanio custodiado por un hilarante Pez Globo armado hasta los nervios. n_n

¿Sabias que los servidores de Hotmail estaban bajo FreeBSD?

Pues así fue los servidores que mantenían los servicios de correo electrónico de Hotmail corrían bajo el sistema operativo UNIX FreeBSD, dicha compañía fue creada por Jack Smith y Sabeer Bhatia, en Julio de 1996, posiblemente eligieron FreeBSD, gracias a su gran rendimiento en cuestiones de escalabilidad, manejo de memoria y seguridad.

Posteriormente en 1997 Hotmail fue adquirida por la Microsoft corp. Quienes hasta Julio del 2000 lograron convertir los servidores de Hotmail de FreeBSD y Apache como servidor Web a Microsoft Windows 2000 Server y como servidor Web Microsoft Internet Information Services 5.0.

Es curioso ver una compañía como Microsoft que apuesta férreamente al uso de patentes de Software y que ha demostrado estar totalmente en contra del Software Libre, adquiera una compañía que hace uso del mismo. Seguramente todo fue por las amplias y jugosas perspectivas comerciales que a futuro implicaba adquirir uno de los sistemas de correo electrónico de mayor auge y repunte del momento. Aun así es morboso ver como el software libre estuvo (nada nos dice que aun no lo este) a disposición y uso de una compañía como Microsoft.

Tomando en cuenta las amplias referencias a favor expresadas por muchos conocedores de la materia sobre la experiencia de utilizar software como FreeBSD y Apache puede dar a pensar que dicha migración fue por cuestiones de orgullo e imagen “Microsoft usando software Libre en un sistema de su propiedad” y el cual le genera enormes ganancias y popularidad entre los consumidores, la popularidad es una ganancia en cierto sentido.

Abra quienes opinen vamos solo fueron aproximadamente dos años, cierto fue un tiempo relativamente corto pero fue un momento en donde la cantidad de usuarios de Hotmail se triplico de pasar de tener una taza de usuarios de aproximadamente 9 millones al momento de la compra a pasar a 30 millones en 1999 un año antes de iniciar la migración. Tiempo en el que tal vez ninguno de los sistemas de red o servidores propiedad de Microsoft, estaban preparados para soportar tal marejada de usuarios que empezaron a abarrotar sus bases de datos. Cabe mencionar que para dichas fechas Microsoft ya disponía de sistemas supuestamente capaces como los de la familia NT así como versiones anteriores del IIS, lo que nos lleva a entender que talvez la idea de que solo fue orgullo e imagen no es tan descabellada cortando así algún posible motivo en contra relacionado con el rendimiento de FreeBSD y Apache.

Curioso ¿no? Software Libre al servicio de Microsoft .-.

Parte de lo aquí descrito puede ser comprobable con el siguiente extracto obtenido del Microsoft TechNET un sitio de índole comunitario propiedad de Microsoft, dedicado a compartir recursos relacionados con las soluciones y proyectos del mismo.

> Microsoft Hotmail web-based e-mail service is one of the most widely adopted free e-mail services available on the Internet. Acquired by Microsoft in 1997, with an installed base of 9 million users, the e-mail service now comprises the largest server farm in the MSN network of Internet services Web properties. The original builders of the application created a two-tier architecture built around various UNIX systems. FreeBSD, a UNIX-like system similar to the Linux operating system, was used to run the front-end Web servers that handled login, Microsoft Outlook Express, and Web-based content delivery tasks. The current network of more than 5,000 servers is organized into about a dozen clusters; each consisting of front-end servers linked to data storage machines.
>
> During June and July of 2000, the Hotmail site was converted from FreeBSD running Apache Web services to Windows 2000 Server running Microsoft Internet Information Services 5.0. The purpose of this technical case study is to describe how that migration was accomplished. The technical case study and the actual process of migration are broken down into three...

distinct stages, specifically: planning, deploying, and operating. The emphasis of this technical case study will be to describe, "How the migration was accomplished" and the factors that were key to its success.

Lo antes descrito solo refleja una pequeña descripción de hechos reales aunados a un poco de reflexión, nada de lo aquí expresado implica, significa o supone una postura real por parte de las compañías aquí mencionadas.

Quiero dedicar este articulo a mi novia Adriana Karmí López Yerena ya que sin élla no hubiera podido terminarlo, gracias mi amor TE AMO.

Enterate de mas:

http://www.oreilly.com/catalog/opensources/book/kirkmck.html

http://www.microsoft.com/technet/interopmigration/case/hotmail/default.mspx

http://es.wikipedia.org/wiki/Hotmail

.FIC | FLOSS International Conference

Congreso Internacional de Software Libre y de Código Abierto

avances científicos del Software Libre en educación tecnología investigación Software Libre y PYMES

Abierto el plazo de inscripción
Más información en **http://softwarelibre.uca.es/fic**
Jerez de la Frontera, España 7-9 de marzo de 2007

Organizadores:

OSLUCA
Oficina de Software Libre de la Universidad de Cádiz

Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Cádiz

SPI&FM
Grupo de Investigación Software Process Improvement and Formal Metods

UCA Universidad de Cádiz
Universidad de Cádiz

Colaboradores:

Ayuntamiento de Jerez
Delegación de Formación y Empleo del Ayuntamiento de Jerez

Escuela de Negocios de Jerez

UCA Universidad de Cádiz Facultad de Ciencias Sociales y de la Comunicación
Facultad de Ciencias Sociales y de la Comunicación Universidad de Cádiz

CEPIS
Council of European Professional Informatics Societies

ati
Asociación de Técnicos de Informática

Patrocinadores:

activa sistemas
Activa Sistemas

Medios Asociados:

gaceta Tecnológica
Gaceta Tecnológica

LINUX MAGAZINE
Linux Magazine

Revista-SL

BSD y sus sabores

opciones muy variadas a tu alcance

Longino Jácome **jacome@php.com.mx**

BSD son las iniciales de "Berkeley Software Distribution" (Distribución de Software Berkeley). Este sistema operativo se deriva del sistema Unix, desarrollado por AT&T en la década de los 70's y modificado a partir de las aportaciones realizadas por la Universidad de California en Berkeley.

En los primeros años del sistema Unix sus creadores, los Laboratorios Bell de la compañía AT&T, autorizaron a la Universidad de California en Berkeley y a otras universidades a utilizar el código fuente y adaptarlo a sus necesidades. Durante las décadas de los setentas y ochentas, Berkeley utilizó el sistema para sus investigaciones en materia de sistemas operativos. Cuando AT&T retiró el permiso de uso a la universidad por motivos comerciales la universidad promovió la creación de una versión inspirada en el sistema Unix utilizando las aportaciones que ellos habían realizado, permitiendo luego su distribución con fines académicos y al cabo de algún tiempo reduciendo al mínimo las restricciones referente a su copia, distribución o modificación.

La última distribución creada por Berkeley fue el BSD 4.4, lanzado en 1995. Desde entonces han aparecido muchas distribuciones basadas en BSD 4.4, tales como FreeBSD, OpenBSD y NetBSD.

La Licencia BSD

Todos los sistemas BSD usan la misma licencia, la cual permite una mayor libertad al usuario, a diferencia de la GPL que es más restrictiva.. El usuario puede hacer lo que desee con el único requisito de que se mencione al autor original. Esto expresa el espíritu abierto y liberal de la comunidad de desarrollo BSD.

Además, la licencia de BSD ha permitido que otros Sistemas Operativos, tanto libres como propietarios incorporaran código BSD. Por ejemplo, Microsoft Windows ha utilizado código derivado de BSD en su implementación de TCP/IP, y utiliza comandos BSD para las herramientas de redes. También Darwin, el sistema en el cual está construido MacOs X, el sistema operativo de Apple, está derivado en parte de FreeBSD 5. Otros sistemas basados en Unix comerciales como Solaris también utilizan código BSD.

El objetivo de este artículo es analizar las diferentes versions de BSD existentes, así como sus características.

OpenBSD - http://www.openbsd.org

OpenBSD es un proyecto que desarrolla un sistema operativo tipo UNIX basado en 4.4BSD, multiplataforma y de libre distribución. Los objetivos principales son la seguridad, las normas, y la portabilidad. OpenBSD no dispone de soporte para más de un procesador. OpenBSD es totalmente libre.

Tanto el código binario como el código fuente son de libre ditribución.

Surgido en 1995, OpenBSD es considerado el Sistema Operativo más seguro del mundo, por lo que es empleado en la instalación de Filtros IP. Al ser su lider Theo de Raadt, canadiense, OpenBSD utiliza software criptográfico que las leyes estadounidenses prohiben exportar.
Caraterísticas

- Funciona en varios tipos diferentes de plataformas de hardware (alpha, amd64, cats, hp300, hppa, i386, mac68k, macppc, mvme68k, mvme88k, sparc, sparc64 y vax).

- Está reconocido por muchos profesionales de la seguridad informática como el sistema operativo tipo UNIX más seguro; esto es el resultado de una intensa e interminable auditoría de seguridad sobre el código fuente.

- Es un Sistema Operativo con todas las funcionalidades de UNIX que se puede adquirir de forma gratuita.

- Integra las últimas tecnologías en seguridad para la implementación de cortafuegos (filtros de IP) y redes privadas virtuales (VPN) dentro de un entorno distribuido.

- Aprovecha el intenso desarrollo actual en muchas áreas, y ofrece la oportunidad de trabajar con tecnologías emergentes dentro de una comunidad internacional de programadores de sistemas y usuarios finales.

FreeBSD - http://www.freebsd.org

FreeBSD es un Sistema Operativo tipo UNIX para plataformas Alpha/AXP, AMD64 e Intel EM64T, i386 IA-64, PC-98 y UltraSPARC basado en el 4.4BSD. También está indirectamente basado en el port de Net/2 para i386 de Berkeley, conocido como 386BSD.

FreeBSD es un Sistema Operativo solido y eficiente. Su principal objetivo es el ser accessible a los usuarios menos tecnicos.

De hecho, FreeBSD es el BSD libre más extendido.

Características:

- Multitarea expropiativa con prioridades dinámicamente ajustadas para asegurar que distintas aplicaciones y usuarios compartan el ordenador de un modo equitativo, incluso bajo la mayor de las cargas.

- La protección de memoria garantiza que las aplicaciones (o los usuarios) no pueden interferirse. Un error fatal en una aplicación no afecta al resto.

- FreeBSD es un Sistema Operativo de 32-bits (de 64-bits en Alpha, Itanium, AMD64, y UltraSPARC) y fue diseñado como tal desde el comienzo.

- Compatibilidad binaria con muchos programas nativos de Linux, SCO, SVR4, BSDI y NetBSD.

- El diseño de la memoria virtual con paginación bajo demanda y de la “caché unificada de VM/buffer” satisface a aplicaciones que requieren grandes cantidades de memoria de forma eficiente aun dando respuestas interactivas a otros usuarios.

- Soporte para SMP en máquinas con múltiples CPUs.

- Extensa documentación en línea.

NetBSD - http://www.netbsd.org

NetBSD es un Sistema Operativo tipo Unix, libre, seguro y altamente portable, disponible para multitud de plataformas desde Opterons a 64-bits y sistemas de escritorio hasta dispositivos de mano y empotrados. Su buen diseño y sus características avanzadas lo hacen excelente para entornos de producción e investigación, además de tener el soporte de los usuarios con el código fuente completo. Hay muchas aplicaciones disponibles, fácilmente instalables.

El slogan del NetBSD es of course it runs BSD, ya que su filosfía es que funcione en el mayor número de plataformas posible. NetBSD es utilizado por muchos grupos de investigación, como el que desarrolla el IPV6 o en la Estación Espacio Internacional.

Características:

- Foco especial en la calidad y portabilidad de código. Portado a 56 arquitecturas.

- Suele ser el pionero en implementar nuevas tecnologías (por ejemplo IPv6).

- Alta seguridad y estabilidad. Fue usado en la NASA.

- Sistema de ficheros BSD FFS (Fast File System), rápido y fiable.

- Seguridad: soporte de IPsec.

- XEN: soporte nativo de máquina virtual XEN en versión 3.0.

DragonFlyBSD - http://www.dragonflybsd.org

DragonFly es un Sistema Operativo derivado de la rama 4.x de FreeBSD. El principal objetivo es tener soporte para multiples procesadores que, en su momento, FreeBSD no tenía.

Actualmente DragonFlyBSD es un sistema operativo robusto, que puede utilizar toda la paquetería desarrollado para la rama 4.x de FreeBSD, con su propio sistema de paquetes y con una imagen ISO que puede ser Live, es decir, no es necesario instalar el so, solo arrancando desde el CD se botea un sistema DragonFlyBSD completo y functional.

GNU-Darwin - http://www.gnu-darwin.org

GNU-Darwin es un proyecto desarrollado a partir del 2000 para crear una distribución libre de Drawin, el núcleo de código abierto de MacOSX.

El objetivo del GNU-Darwin es ser una Distribución libre de Darwin, reemplazando las aplicaciones nativas de Apple por aplicaciones open source. GNU-Darwin funciona en cualquier computadora que oficialmente soporte el MacOSX, así como en algunos equipos INTEL.

El proyecto NetBSD

un BSD que funciona en una tostadora

César Yáñez Fernández **hokum@e-compugraf.com**

¿NetBSD? ¿Otra "distribución de BSD? ¿Cuántas hay? - Son las preguntas más comunes cuestionadas por los grupos de usuarios de Software Libre que no ha escuchado acerca de éste Sistema Operativo; en este texto se responderán a éstas y otras preguntas, además de formularse otras nuevas conforme se vaya leyendo más adelante.

NetBSD es un proyecto internacional el cual pretende desarrollar, mantener y mejorar un Sistema Operativo "UNIX-like" altamente portable e interoperable, proveniente del Sistema Operativo UNIX BSD (Berkeley Software Distribution), desarrollado alguna vez por el Grupo de Investigación en Ciencias Computacionales (CSRG) de la Universidad de California en Berkeley. NetBSD no sólo es el proyecto de desarrollo del Sistema Operativo con el mismo nombre, también es una Fundación sin ánimos de lucro que protege legalmente a éste.

NetBSD toma su nombre como herencia de ése Sistema Operativo BSD UNIX y de Internet en forma de hacerle tributo a su modelo de desarrollo, el cual es una comunidad de programadores de todo el mundo, comunicados a través de la supercarretera de la información, el mismo modelo de desarrollo que se ha usado desde los primeros días de Linux, de BSD del CSRG y de 386BSD; el cual fue el primer Sistema Operativo UNIX-like libre que se ejecutó en los equipos PC con procesador Intel 386, el primer modelo de procesadores de la arquitectura Intel compatible que operaba a 32 bits.

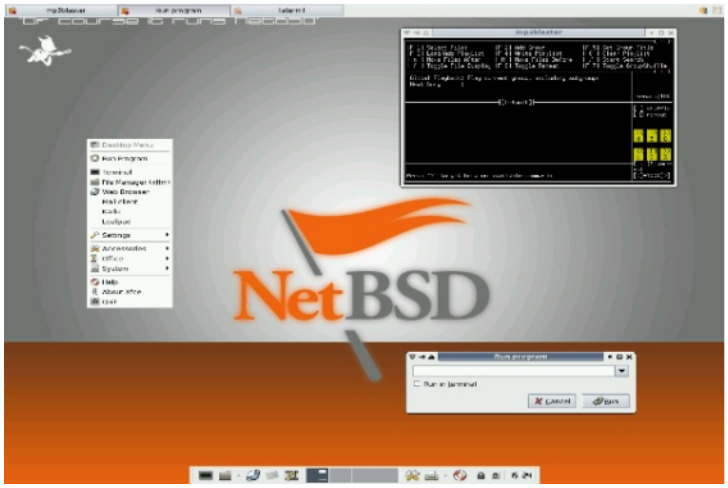
NetBSD viendolo como proyecto de desarrollo, tiene como objetivos el tener un sistema de desarrollo planeado y evolucionario, además de enfocarse en tener un control de calidad y mejora continua dentro del proceso de desarrollo. Otra característica que tiene NetBSD la cual lo ha hecho importante es su amplio soporte de plataformas de Hardware: 59 tipos de computadoras entre 14 arquitecturas de procesadores diferentes; algunos podrían incluso comentar de que NetBSD corre en más plataformas de hardware que Linux. Otro punto que hay que recalcar del proyecto NetBSD es su licencia BSD, la cual es una licencia justificada como Libre ante la Free Software Foundation, y que ésta licencia permite también el hacer modificaciones al programa y que éstas modificaciones no tienen que ser necesariamente libres; ésta característica a la licencia BSD puede provocar confusión y también crea estragos en la comunidad de Software Libre.

El Sistema Operativo NetBSD cuenta principalmente de 4 partes: el sistema para compilación, el cual es el set de GNU; el núcleo (kernel); el software de sistema y de usuario; y el árbol de ports, conocido como pkgsrc. Con éstos componentes, NetBSD asegura que es Software Libre. Otro objetivo de NetBSD es hacer que todos sus componentes, incluyendo el sitio web oficial del proyecto, se encuentren disponibles en Internet, tanto en formato binario como su código fuente, a través del sistema de control de versiones CVS.

La seguridad dentro del proyecto NetBSD tampoco se hace a un lado, ya que cuenta con un proceso de auditoría al código fuente línea por línea, y mecanismos de control mediante sysctl.

Todo el desarrollo de NetBSD es resguardado por la Fundación NetBSD, una organización sin ánimos de lucro, instalada en Delaware, E.U., la cual está dirigida por una mesa de directores, elegidos por medio de votos de la comunidad de desarrolladores de NetBSD; la cual, es dueña de los servidores del proyecto, maneja el control de donaciones de dinero, servicios, hardware y también tiene copyright sobre NetBSD y su marca registrada.

NetBSD es todo un mundo organizado y que sigue los ideales del Software Libre para mejorar la tecnología sin barreras ni golpes.



NETBSD 4.0

NetBSD 4.0 es la siguiente versión importante de NetBSD, aún sin definir su fecha final de lanzamiento a disponibilidad. Al igual que las vesiones importantes anteriores (la serie, 3.x, 2.x, 1.x) contiene una lista considerable de cambios y nuevas tecnologías agregadas a NetBSD, entre las que más se destacan:

- Soporte para nuevos dispositivos SATA.
- Mejora de soporte de TCPv6 (no confundir con IPv6).
- Uso de GCC-SPP para compilar el Sistema Operativo.
- Soporte para TMPFS.
- Soporte para nuevos dispositivos WiFi.
- Soporte para PufFS.
- Nuevo sistema de Runlevels al estilo SysV (no confirmado).
- Soporte para Xen 3.x
- Soporte para Bluetooth.
- Nuevo sistema HAL (Hardware Abstraction Layer).
- Soporte para CARP (Common Address Redundancy Protocol).
- Soporte multi-usuarios en entornos chroot.
- Soporte para NDIS Wrapper.
- Removido soporte de Kerberos 4.
- Nuevo soporte de Framebuffer VESA para PC.
- Agregado pantalla BootSplash sobre Framebuffer.
- La versión más nueva de PF.
- Mejoras a LFS.
- Nuevo sistema de PAM.

Tal vez para los usuarios de Linux lectores de ésta revista, éstas tecnologías nuevas a lo mejor ya existan desde hace rato en Linux e incluso en FreeBSD; pero hay que recordar que el proyecto NetBSD tiene un grupo de desarrolladores pequeño y que no busca ser innovador, sino un producto de calidad.

Sitio oficial de NetBSD:

http://www.netbsd.org

El proyecto PC-BSD
la informática personal al estilo bsd

Gonzalo Martínez **g.martinez@pcbsd.es**

Este artículo pretende informaros, sin comentarios excesivamente técnicos, sobre este nuevo Sistema Operativo, cuyo uso ofrece claras ventajas sobre los demás sistemas. Os hablaré de sus caracterásticas y de los puntos en los que se diferencia del resto. Seguiré un orden cronológico de las dudas y preguntas que suelen plantearse al usar por primera vez este sistema, una vez instalado.

Pero antes de comenzar y a modo de presentación, dejádme deciros que PC-BSD, a pesar de ser un Sistema Operativo muy reciente, concretamente la primera versión data de Abril del 2005, su aceptación en el mercado es excelente y cuenta ya con miles de adeptos y seguidores. Esto es debido, principalmente, a que es un proyecto con valores y metas exclusivas, ya que su objetivo no es otro que cambiar el chip a esas personas que todavía relacionan *BSD con los servidores y no aceptan que pueda usarse para escritorio.

Tambien va dirigido a todos aquellos que piensan que Unix/Linux son sólo para gurús de la informática y para friquis.

PC-BSD quiere perfeccionar, el ya muchas veces denominado mejor Sistema Operativo: FreeBSD, para que sea accesible por cualquier persona independientemente de su conocimiento sobre informática. Su intención es hacer un Sistema Operativo más sencillo, más claro, más amigable y más fácil de usar, sin que se pierdan puntos en su seguridad y rapidez.

Uno de los primeros logros que se ha conseguido es la sencillez a la hora de instalarlo. El instalador es totalmente gráfico e intuitivo. Nada tienen que ver con las Opciones Avanzadas, (éstas se quedan para los que tienen tiempo para perderlo) y se centra directamente en tus necesidades. De manera que la primera pregunta que debes hacerte es ¿para que quiero usar PC-BSD?... ¿Lo quiero como servidor?....¿Lo quiero como escritorio?.... Y según la opción que elijas, y de forma casi mágica para el usuario, se instala lo esencial para uno u otro. El sistema simplifica los pasos a seguir para aquellos que quieren instalarlo y tenerlo rápidamente listo para ser usado, sin exterminar las opciones un poco mas complejas y mucho menos simplificadas que algunos usuarios podrían echar de menos.

Por defecto, PC-BSD también prepara el ordenador como desktop productivo y listo para la acción desde el primer reinicio, instalando numerosas herramientas esenciales para este objetivo:

Un Entorno Gráfico KDE

Echándome a un lado de la guerra dialéctica típica de escritorios, KDE si no es el mejor, sin duda, es de los mejores. Y desde luego, el más claro para usuarios inexpertos. KDE es un proyecto muy prometedor, con muchísimas opciones dentro del SoftWare Libre y que cuenta ya con muchísima aceptación.

Soporte de idiomas

Es un handicap hoy en día, y es increible que existan tantas distribuciones o Sistemas Operativos tan personalizados y que solo hablen

en un idioma , olvidando al resto. Traemos la opción de poner PC-BSD en cincuenta idiomas.

Actualizador online

Herramientas de actualización a través de internet de forma automática, aunque tambien tienes la opción de actualizarte sin conexión a internet para los que se encuentren ante el gran problema de no poder utilizar la red, bajándote la actualización.

Gran variedad de opciones disponibles:

1.- Mediante la utilización del PBI

Este concepto es fácil de entender y necesario. PBI significa Pc Bsd Installer o Instalador PcBsd. Pues bien, los PBIs engloban unas mejoras muy importantes que permitirán en un futuro cercano que este Sistema Operativo consiga su propósito: llegar a todo tipo de usuarios.

Esta afirmación tan rotunda la hago porque he comprobado que el Sistema PBI es el único que puede facilitar al usuario la instalación de un programa de forma rápida y sencilla.

El proceso es el siguiente: Debes ir a la página oficial que soporta los paquetes pbi: http://www.pbidir.com

Después, buscar el programa que te interesa instalar...

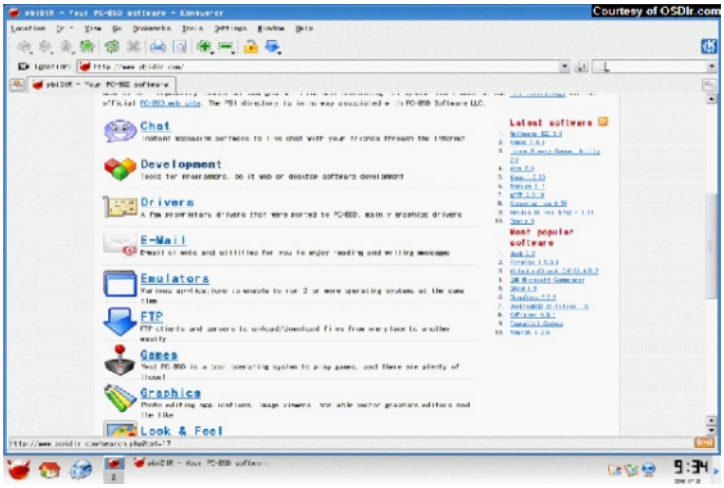
Una vez descargado éste, debes hacer doble click sobre él para leer las instrucciones y la Licencia de Uso del Programa.

Finalizada la instalación, en el Menú de KDE te aparecerá PBI Programs o Programas PBI y allí verás el programa instalado, con un enlace a su binario, a su página web y a su manual online, si es que lo tiene.

Al instalarlo, tambien aparecerá la nueva entrada en el Menú para desinstalar programas.

Y si quieres desinstalar el programa, simplemente seleccionas cuál de ellos quieres desinstalar y le das al botón de Eliminar.

Se eliminará de forma automática y sin dejar absolutamente ni rastro.



El formato PBI procura facilitar al máximo la relación usuario-software para que nunca tenga que usar la consola. Tambien los PBIs engloban una manera de distribuir el Sistema Operativo, ya que todo se almacena en una misma parte del Sistema Operativo, todo se encuentra instalado en el directorio Programs o programas del ordenador, facilitando mucho las cosas a la hora de buscar un programa o una aplicación.

Una vez explicado que es un PBI, ahondará todavía más en este punto para aquellos que quieren saber como funciona un PBI. Con el PBI lo que se hace es recoger todo el programa con todas sus librerías y dependencias. Esos binarios tienen que saber que las librerías que buscan están en su mismo directorio. El apuntar al directorio no es complicado, lo complicado es cuando la aplicación requiere de otras pequeñas aplicaciones o pequeñas librerías para ser ejecutadas, las cuales tendrán que ser añadidas.

El sistema PBI lo que hace es "enjaular" el programa, incluido en memoria. Es decir, el PBI incluso no compartirá las librerías. Esta forma de actuar tiene sus defensores y detractores .

La gente que no está deacuerdo con esta forma de hacer se basan en la idea de que, al no compartir las librerías, se ocupará el doble de memoria RAM.

Y esta cuestión, aunque sea cierta, no es la ideal ya que preferimos ocupar el doble de Kb, teniendo en cuenta como avanza la informática y viendo ordenadores que por defecto traen 2 Gb de memoria RAM. Es preferible ocupar más memoria, si con ello estamos seguros de que el programa que nos hemos bajado va a funcionar correctamente.

Os preguntaréis que por qué no va a funcionar correctamente un programa por el simple hecho de que las librerías están compartidas.

Bueno, pues ese es el problema que actualmente tiene Linux.

Si tienes un programa que utiliza una librería concreta que se comparte con otro programa, el primero, en un momento determinado puede que te diga que quiere actualizar esa librería, y el segundo programa en ese momento, puede que no te diga nada. Es más, puede que el quiera la librería antigua no la nueva. Y entonces, no se tiene más remedio que tomar una decisión, porque es imposible usar ambos programas como antes si podía.

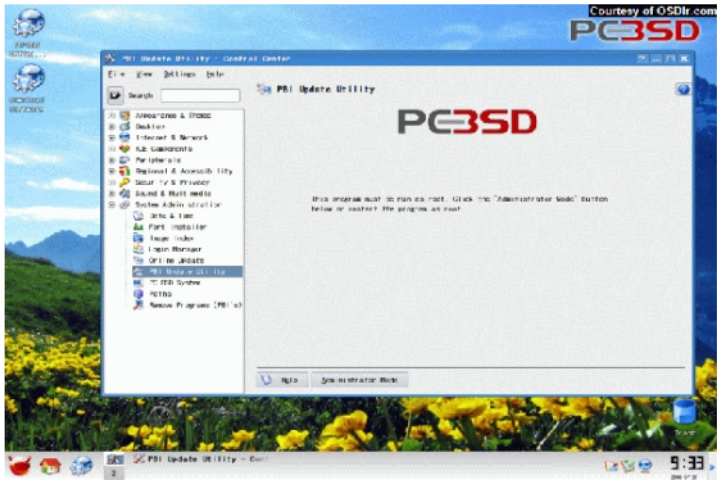
Yo particularmente lo último que probe de Linux fue una versión de Ubuntu, la cual me gustó mucho hasta que tuve problemas con las librerías. En un momento determinado el programa entró en barrena pidiéndome constantemente que arreglara las librerías rotas.

Los PBis nunca podrán tener este tipo de errores.

2.- Mediante los Ports de FreeBSD

Siendo fieles a la base de nuestro Sistema Operativo, PC-BSD mantiene el sistema de ports de FreeBSD. Los ports son directorios en nuestro ordenador que automáticamente apuntan, descargan e instalan/compilan las aplicaciones que se quieren utilizar. Tampoco querría adentrarme mucho en este mundo ya que está muy bien tratado en otro artículo de vuestra revista.

Simplemente remarcaré esta opción como otra posibilidad. Una opción que actualmente ofrece más de 14000 aplicaciones para ser instaladas.



3.- Paquetes tarballs

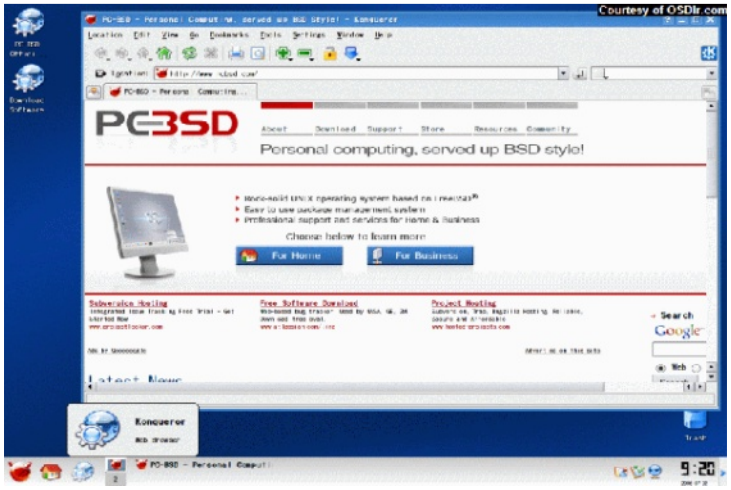
Simplemente los nombraré para decir que si te bajas el tarball de la aplicación, con un pkg_add puedes instalarlo en tu Sistema Operativo. Se mantiene en estos dos últimos, la manera más clásica de instalación que nos ofrecía FreeBSD

Ahora que ya sabemos qué formas hay de instalar paquetes, surgiran las dudas de si todos los programas funcionaran en PC-BSD. Bien, pues hasta ahora vamos a la par que con Linux. Todo programa que sea nativo en Linux, puede correr en PC-BSD, usando la compatibilidad con Linux.

Por contra, sólo corremos los programas para Windows que sean capaces de utilizarse bajo el emulador de Windows llamado Wine.

De todas maneras es lo de siempre, todo programa para Windows tiene su clon para el resto de Sistemas Operativos, el paso que tiene que dar la gente es superar ese concepto y aceptar que los programas homonimos funcionan igual o mejor que los que nos venden casas como Adobe Macromedia o Microsoft.

Yo creo que pocas dudas quedan ya, con lo cual pasaré a sobrevolar el siguiente punto que considero de interés:



Espectativas de PC-BSD

Este punto es más una nota informativa. Quisiera remarcar que PC-BSD está creciendo a un ritmo muy bueno y más ahora tras la reciente compra de iX Systems de nuestro Sistema operativo, lo cual permitirá una dedicación de los programadores del 110% a la depuración del código.

A su vez dará una inyección de vitalidad, mayor difusión, estable intrusión en el mundo comercial e intentará ser el punto de partida de una escalada hacia lo más arriba.

iX Systems ahora puede presionar de manera más eficaz al resto de empresas para que seamos tomados enserio y por ejemplo, que saquen un plugin de Flash nativo para PC-BSD.

Pronto, para dentro de unas pocas semanas esperamos tener la versión 1.3 preparada para su publicación, donde instalaremos lo último de lo último en cuestion de aplicaciones y del escritorio y habremos depurado almenos unas mil lineas de código.

A pesar de todo, el único que puede deciros si PC-BSD merece la pena o no, si es un buen proyecto o no, sois vosotros. Os diría que lo probeis y vosotros mismos juzgueis.

Si deseas obtener mas información sobre PC-BSD, los desarrolladores del proyecto te invitan a visitar los siguientes sitios web;

http://www.pcbsd.org
http://www.pbidir.com
http://www.pcbsd.es

XGL, AIXGL, COMPIZ & BERYL
la revolución gráfica en gnu/linux

Gonzalo González **gonzalo.gonzalez@revista-sl.org**

Desde el 2 de enero del 2006, la revolución gráfica en GNU/Linux ya es una realidad esto gracias al lanzamiento al público del Xgl por parte de David Reveman (Novell). Pero fue hasta un mes después que Novell realizó una demostración de Xgl, precisamente en el XDEVCONF 2006 realizado en el "Sun Microsystems Santa Clara (Agnews) Campus" en Santa Clara, California, E.U.A., atrayendo la atención de muchos.

El motivo de la existencia de Xgl es la de tener un sevidor X completo, utilizando OpenGL para comunicarse con el Hardware y cualquier aplicación (X11 u OpenGL) pueda disfrutar el poder de la aceleración por hardware. Actualmente existen dos implementaciones de Xgl: Xglx y Xegl.

Xglx es la implementación actual que corre sobre X.org utilizando las extensiones GLX como la comunicación entre estos dos servidores X, así le daría a las aplicaciones X11 renderización indirecta y a las aplicaciones OpenGL renderización directa.

La implementación Xegl es un completo servidor X, ya no dependerá de X.org, que seguirá usando OpenGL para escribir directo a frammer buffer de Linux.

Debido a que el desarrollo de Xgl representa un largo trabajo por tratarse de la programación completa de un servidor X, Red Hat y la fundación X.org plantearon modificar ligeramente X.org para que soportara la aceleración indirecta utilizando las extensiones GLX y los drivers DRI. A este proyecto se le llama Aiglx.

Aunque parezca un poco ilógico, la combinación Xorg + Xglx consumen menos memoria que Aiglx, esperemos que por el momento.

Esta nueva evolución de los servidores X nos proporciona las herramientas para poder aprovechar el máximo posible de la aceleración por hardware de nuestras tarjetas gráficas. Pero necesitamos algunos elementos más para tener un escritorio que aproveche estas herramientas como lo son el Gestor de Ventanas y el Gestor de Composición. El primero, controla la ubicación y apariencia de la ventana (posición y tamaño, respectivamente). Mientras el que se encarga de dibujar las ventanas es el gestor de composición.

Desde que Keith Packard escribio la X-extensión "composite" que dibujaba todas la ventana, aun cuando esta no era visible en una parte o totalmente, a este modelo de composición se le llama off-screen (fuera de pantalla).

Compiz llegó al mundo Xgl, de la mano de Dave Reveman, para ser el Gestor de ventanas y de composición para, en un principio, sustituir al Metacity de Gnome. Aunque tiene una clara dependencia con Gnome (utiliza Gconf para cargar sus efectos), puede sustituir al kwin de KDE. Compiz funciona en base a plugins, lo cual permite agregar nuevos efectos como lo es la representación de cada escritorio virtual en un lado de un cubo.

Quinnstorm realizó un fork de Compiz llamandolo Beryl, esto debido a que Quinnstorm y el equipo de desarrollo de Compiz no tuvieron un acuerdo en la adopción de los cambios realizado en Compiz por Quinnstorm en el Compiz-vainilla.

El 19 septiembre del 2006 fue anunciado oficialmente.

Entre las características más notables de Beryl sobre Compiz son:

- El decorador de ventanas, conocido como cgwd, ahora es llamado emerald
- Utilización de un archivo plano en vez de Gconf.
- Gestor de temas llamado emerald-themer.
- Gran variedad de plugins.

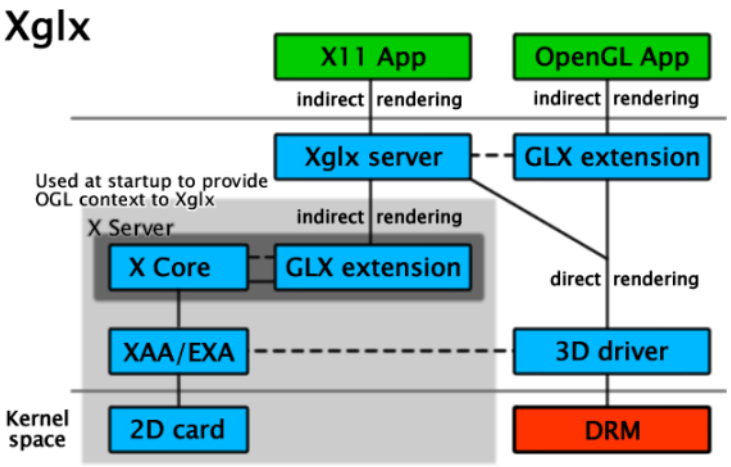
Una pequeña forma de explicar la interconexión de estas aplicaciones es:

Display – Xorg + Xglex/ Aiglx / Xegl – Compiz / Beryl - Desktop

En mi poca experiencia con estos nuevos juguetes, he podido disfrutar estas dos combinaciones:

Xgl + Compiz y Xgl + Beryl.

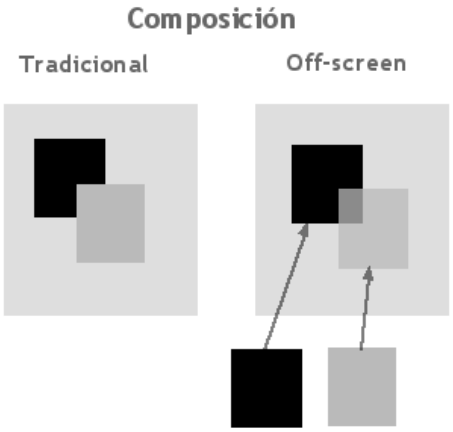
En el primer caso, fue en una computadora con características Pentium 4 a 2 Ghz, Placa base D865GBF, 512 MB en Ram y monitor Samsung Syncmaster 550v. La distribución elegida fue la de OpenSuse 10.2. El único problema que tuve fue la de configurar bien el X.org para que la tarjeta de video integrada pueda soportar la aceleración 3D ya que los paquetes necesarios ya estaban instalados por defecto. Quedando la sección "Device" y "Monitor"de mi xorg.conf de la siguiente manera:



Comunicación con Xglx

```
Section "Device"
  BusID         "0:2:0"
  Driver        "i810"
  Identifier    "Device[0]"
  Option        "NoDDC"
  Screen        0
  VendorName    "Intel"
  Videoram      65536
EndSection

Section "Monitor"
  Option        "CalcAlgorithm" "XServerPool"
  HorizSync      28-49
  Identifier    "Monitor[0]"
  Option        "DPMS"
  VendorName    "SAMSUNG"
  VertRefresh    43-72
  UseModes      "Modes[0]"
EndSection
```



Diferencias en la composición

Luego que nos aseguramos de que ha arrancado el servidor X correctamente, entramos al centro de control de OpenSuse, luego click en efectos de escritorio y nos aparecerá una ventana la cual tendrá un botón con el texto: Activar efectos de escritorio. Un click sobre este botón y se activarán los efectos. Todos los efectos trabajan bien el problema es que se siente un poco lento.

El otro caso, Xgl + Beryl.

Esta vez en una computadora Pentium 4 HT a 3 GHz, placa base d101ggc, 512 MB en Ram y Monitor Samsung Syncmaster 540n (lcd de 15"). Las distribución: Ubuntu 6.06.

Las instrucciones para instalar Xgl+Beryl están en la sección "enlaces" de este artículo.

Como siempre el único inconveniente es la configuración de la tarjeta de video y el monitor:

```
Section "Monitor"
        Identifier   "SyncMaster-540n"
        HorizSync    30.0 - 61.0
        VertRefresh  56.0 - 75.0
        Option      "DPMS"
EndSection

Section "Device"
        Identifier      "ATI Technologies,
Inc. Radeon Xpress 200 (RC410)"
        Driver          "fglrx"
        Option          "VideoOverlay" "on"
        Option          "OpenGLOverlay" "off"
        BusID           "PCI:1:5:0"
        Option          "UseInternalAGPGART"
"no"
        Option          "KernelModuleParm"
"locked-userpages=0"
        Option          "Capabilities"
"0x00000800"
        Option          "UseFastTLS" "off"
        VideoRam        65536
EndSection
```

¿Lo bueno? muy rápdo, muchos plugins y eye candy.

¿Lo malo? Después de un buen rato de aprovecharme al máximo de la tarjeta de video, lease como una sesión de 3 a 5 Hrs de Xgl+Beryl, se me cierra la sesión.

Los dos casos tienen sus detalles, lento y el cierre de sesión respectivamente, pero quiero creer que es por cuestión de alguna configuración extra. Cuando halla más tiempo trataré de corregir eso detalles y publicarlos.

Espero que usted, amable lector se anime a experimentar la nueva revolución de gráficos en GNU/Linux y por que no, en otros Sistemas Operativos Libres.

Para obtener mas información se recomienda visitar estos enlaces:

http://gentoo-wiki.com/HARDWARE_Video_Card_Support_Under_XGL

Beryl on Ubuntu - http://crysol.inf-cr.uclm.es/node/399

http://www.freedesktop.org/wiki/Software/Compiz

http://www.beryl-project.org/

http://principe.homelinux.net/

http://wiki.x.org/wiki/XDevConf

http://es.wikipedia.org/wiki/Xgl

http://www.tuxpan.com/fcatrin/es/comments.php?guid=20060311

Un vistazo a OpenBSD

sistema operativo seguro por naturaleza

Carlos Augusto Lozano **carlos.augusto@revista-sl.org**

OpenBSD es un Sistema Operativo tipo UNIX basado en 4.4BSD y NetBSD que surgio debido a las diferencias filosoficas del lider del proyecto Theo de Raadt con los demas miembros del proyecto NetBSD.

OpenBSD tiene un especial énfasis en la segurdad, para cumplir con sus propositos OpenBSD se concentra en la portabilidad (una de las herencias de NetBSD), criptografía integrada, correción y seguridad proactiva.

Algo muy caractéristico de OpenBSD es su humor sarcástico, este se expresa en los CDs oficiales de OpenBSD, decorados con caricaturas alusivas al proyecto y sus ventajas frente a otros sistemas operativos.

Al momento de escribir estas lineas, la versión estable mas actual es la 4.0, liberada el 1ero de Noviembre del 2006.

Sin adentrarnos tanto en historia, caraceteristcas u otros detalles que facilmente podrián encontra en la página web del proyecto; me concentrare en mostrarles la forma de dar el primer paso para hacer uso de este sistema operativo. Me refiero a la instalación la cual a primera vista puede parecer complicada, pero una vez que se realiza resulta ser de lo mas sencilla.

Instalando OpenBSD

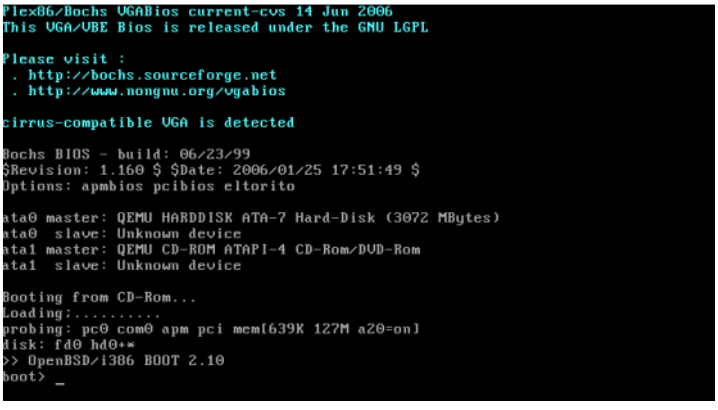
Ya sea que obtengamos el sistema operativo adquiriendo un CD oficial, mediante una imágen ISO no oficial o descargando las fuentes y creando una nosotros mismos, los pasos serán los mismos.

La primer parte tiene que ver con el "booteo" del CD, en donde se reconocera nuestro hardware, y podremos seleccionar opciones sobre la distribución del teclado, así como la unidad de disco duro en donde se instalrá el sistema.

Una vez que haya bootedo completamente el CD, la primer pregunta que leeremos, será "¿Que desea hacer?: (instalar, actualizar o usar la linea de comandos)". En este articulo nos centraremos en la instalación, la actualización es muy sencilla, aunque hay que recalcar que lo mas recomendable son las actualizaciones de versiones continuas, es decir 3.9 a 4.0, etc.

Al seleccionar "Insalación" nos preguntara sobre nuestra distribución de teclado y unidad de disco duro donde planeamos realizar la instalación.

Muy importante es leer cada uno de los avisos, pues en este caso, por ejemplo, nos preguntara si deseamos hacer uso del disco duro completo o solo una parte.



Una vez que hayamos definido la unidad en destino de la instalación deberemos de crear las diferentes particiones que compondrán OpenBSD.

Son varias las particiones, recomendadas, para la instación, además de tener también un tamaño dependediendo de cada una. Esta es la lista de particiones con sus tamaños recomendados:

- / (80 MB)
- Swap (300MB, aunque depende de la cantidad física de memoria RAM)
- /tmp (80 MB)
- /var (80 MB)
- /usr (2 GB)
- /home (todo el espacio que dispongamos, pues será el directorio usado para los directorios de trabajo de los usuarios del sistema)

Una vez creadas las particiones pasaremos por un proceso de verificación de el tamaño y punto de montaje de cada una, para poder revertir cualquier cambio que fuese necesario. Una vez que hayamos decidido que estas son correctas se acepta la escritura de las nuevas particiones en el disco duro.

```
  b:       614880       164304    swap
  c:      6291456            0  unused      0     0
  d:       164304       779184  4.2BSD   2048 16384   16 # /tmp
  e:       164304       943488  4.2BSD   2048 16384   16 # /var
  g:      4194288      1107792  4.2BSD   2048 16384   16 # /usr
  h:       987840      5302080  4.2BSD   2048 16384   16 # /home
> w
> q
No label changes.
Mount point for wd0d (size=82152k)? (or 'none' or 'done') [/tmp]
Mount point for wd0e (size=82152k)? (or 'none' or 'done') [/var]
Mount point for wd0g (size=2097144k)? (or 'none' or 'done') [/usr]
Mount point for wd0h (size=493920k)? (or 'none' or 'done') [/home]
Mount point for wd0d (size=82152k)? (or 'none' or 'done') [/tmp] done
No more disks to initialize.

OpenBSD filesystems:
wd0a /
wd0d /tmp
wd0e /var
wd0g /usr
wd0h /home

The next step *DESTROYS* all existing data on these partitions!
Are you really sure that you're ready to proceed? [no] _
```

A continuación sigue la configuración de los aspectos relacionados a la red, como el nombre de host, dispositivos de red, asignación de una IP o uso del servicio DHCP.

Ademas vendra la definición de la contraseña del administrador del sistema, la cual desde este momento es sumamente importante pues OpenBSD no añade usuarios durante la instalación.

Una vez que hayamos terminado de definir estos puntos se procede con la selección del medio de instalción, el cual puede ser el CD desde el cual hemos arrancado la instalación o algun repositorio en Internet.

```
        [X] comp40.tgz
        [X] man40.tgz
        [X] game40.tgz
        [ ] xbase40.tgz
        [ ] xetc40.tgz
        [ ] xshare40.tgz
        [ ] xfont40.tgz
        [ ] xserv40.tgz
Set name? (or 'done') [bsd.mp] *

        [X] bsd
        [X] bsd.rd
        [X] bsd.mp
        [X] base40.tgz
        [X] etc40.tgz
        [X] misc40.tgz
        [X] comp40.tgz
        [X] man40.tgz
        [X] game40.tgz
        [X] xbase40.tgz
        [X] xetc40.tgz
        [X] xshare40.tgz
        [X] xfont40.tgz
        [X] xserv40.tgz
Set name? (or 'done') [done] done_
```

Ya que se haya tomado una desicion sobre desde donde se realizara la instalación, definiremos los "sets" a instalar. Por defecto el instalador nos selecciona los mas básicos, pero esta selección puede ser modificada por el usuario. Una vez que hayamos seleccionado los sets podemos aceptar y el sistema empezará su instalación.

Al finallizar solo faltara definir algunos parametros como los servicios activos, si deseamos usar el sistema X Window, el uso horario, etc. Ya finalizada la instalación aparecera un mensaje de felicitaciones y agradecimientos por usar OpenBSD.

```
Brazil/     Etc/        Hongkong    MST7MDT     ROC         posix/
CET         Europe/     Iceland     Mexico/     ROK         posixrules
CST6CDT     Factory     Indian/     Mideast/    Singapore   right/
Canada/     GB          Iran        NZ          Turkey      zone.tab
What timezone are you in? ('?' for list) [Canada/Mountain] Mexico
What sub-timezone of 'Mexico' are you in? ('?' for list) ?
BajaNorte  BajaSur    General
What sub-timezone of 'Mexico' are you in? ('?' for list) General
Setting local timezone to 'Mexico/General'...done.
Making all device nodes...done.
Installing boot block...
boot: /mnt/boot
proto: /usr/mdec/biosboot
device: /dev/rwd0c
/usr/mdec/biosboot: entry point 0
proto bootblock size 512
/mnt/boot is 3 blocks x 16384 bytes
fs block shift 2; part offset 63; inode block 216, offset 2984
using MBR partition 3: type 166 (0xa6) offset 63 (0x3f)
done.

CONGRATULATIONS! Your OpenBSD install has been successfully completed!
To boot the new system, enter halt at the command prompt. Once the
system has halted, reset the machine and boot from the disk.
# _
```

Desde este momento ya podemos usar OpenBSD, sin embargo algo que considero muy importante para las personas que usan por primera vez este tipo de sistemas operativos, es contar con un ambiente grafico desde el cual ejecutar aplicaciones que comunmente usamos en sistemas operativos como GNU/Linux o BSDs mas amenos al usuario de escritorio como PC-BSD o BSD Desktop.

Por ello también deseo explicarles la forma de instalar algún escritorio.

Gnome desde ports

Los "ports" son un conjunto de archivos make usado por los sistemas BSD para instalar software de forma sencilla curbriendo dependencias.

El "Árbol de ports" se encuentra en un directorio en OpenBSD, para instalar un escritorio, en este caso Gnome, solo debemos dirigirnos a la ruta:

```
#cd /usr/ports/x11/gnome
```

Una vez alli basta con usar el comando make para crear el binario.

```
#make
#make install
```

Y listo, desde ese momento tendremos Gnome instalado en nuestro sistema. Ahora se debe de incluir en el archivo .xinirc gnome-session y de esa forma cada que tecleemos:

```
$startx
```

Iniciará Gnome.

Una vez que se tiene un ambiente gráfico es mas senicllo empezar a aprender mas cosas sobre OpenBSD.

Existen muchos recursos, como la documentación oficial, algunos eBooks en las redes P2P, etc.

También podrán encontrar mas información en sitios web, aunque lamentablemente la mayoria de los sitios sobre BSD en español ponen atención a otros BSDs, sin embargo en sitios web en ingles encontrarán mucha mas información.

Para obtener mas información pueden visitar:

http://www.openbsd.org

Desarrollo de aplicaciones Web

desarrollo con LAMP.NET (2/3)

Martin Marquez **xomalli@gmail.com**

Conexión a PostgreSQL usando Npgsql

Para conectarse a una base de datos PostgreSQL es necesario tener instalado el controlador (driver) Npgsql, el cual puede usarse descargándolo del sitio http://pgfoundry.org/projects/npgsql/ o bien asegurarse de que este incluido en la instalación de Mono, recomiendo descargar la última versión de Mono la cual incluye las últimas versiones del controlador.

Probar la conexión a la base de datos

Antes de usar el controlador vamos a crear la base de datos, suponiendo que el usuario con el que trabajamos tiene los privilegios necesarios, usamos el siguiente comando:

```
$ createdb inventario
```

Un programa que nos ayudará a probar la conexión a la base de datos.

```
using System;
using Npgsql;

namespace classes.computo{
     class PCon{
          static NpgsqlConnection conn =
null;

          public static int Main(string[]
args){
               try{
                    Console.WriteLine("Proban
do la conexión a PostgreSQL...");
                    conn = new
NpgsqlConnection();
                    conn.ConnectionString =
@"Server=127.0.0.1;Port=5432;Database=inventa
rio;User ID=postgres;Password=chikoaze";
                    conn.Open();
    Console.WriteLine("Conexion abierta");
                    return 0;
                    }catch(Exception e){
                     //Console.WriteLine(e.T
oString());
                    Console.WriteLine("Sin
Conexion");
                    return -1;
               }
          }
     }
}
```

Creación de las tablas

Teniendo probada la conexión, ahora crearemos un guión SQL para la base de datos, primero empezaremos por los catálogos de nuestro formulario principal y al final la tabla del formulario principal.

```
Create table OperatingSystems(
id serial PRIMARY KEY,
name varchar NOT NULL,
created_on timestamp NOT NULL default
current_timestamp,
updated_on timestamp NOT NULL  default
current_timestamp
);

CREATE table Brands(
id serial PRIMARY KEY,
name varchar NOT NULL,
created_on timestamp NOT null  default
current_timestamp,
updated_on timestamp NOT null  default
current_timestamp
);


Create Table Equipment (
id serial PRIMARY KEY,
```

```
serialnumber varchar,
brand_id integer NOT NULL references
Brands(id) On Delete Cascade On Update
Cascade,
operatingsystem_id integer NOT NULL
references OperatingSystems(id) On Delete
Cascade On Update Cascade,
rammemory varchar,
hardDisk varchar,
hostname varchar,
created_on timestamp NOT null  default
current_timestamp,
updated_on timestamp NOT null  default
current_timestamp
);

/*Para llenar los catalogos*/
INSERT INTO
OperatingSystems(name)VALUES('Linux SuSe
10');
INSERT INTO
OperatingSystems(name)VALUES('Windows XP');
INSERT INTO Brands(name)VALUES('IBM');
INSERT INTO Brands(name)VALUES('HP/Compaq');
```

Programando los formularios WEB

Las novedades de .NET para la programación de formularios Web, es la capacidad de poder programar los formularios Web usando los controles HTML pero con la posibilidad de ejecutarlos del lado del servidor y los controles Web (Web Controls), para estas formas es necesario programarlos usando la técnica (Code Behind) es decir teniendo un archivo vista con código vista HTML (.aspx) pero enlazado a un archivo controlador de clase en C# (.cs).

Usando controles HTML del lado del servidor (HTMLControls)

Usaremos primeramente los HTMLControls para programar el formulario principal, el siguiente código es el formulario principal con las modificaciones necesarias para el uso de los HTMLControls.

```
<%@ Page language="c#"
Codebehind="registro.aspx.cs"
Inherits="computo.classes.registro"%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
  <title>Formulario de registro de
computo</title>
</head>
<body>
<FORM name="forma1" method="POST"
runat="server">
<table>
<TR>
        <TD>Numero de serie</TD>
        <td><INPUT type="text"
name="serialnumber" id="serialnumber"
runat="server"></td>
</TR>
<tr>
        <TD>Marca</TD>
        <td><%=marcas%></td>
</tr>
<tr>
        <TD>Sistema Operativo</TD>
        <td><%=operativos%></td>
</tr>
<tr>
        <TD>Memoria principal</TD>
        <td><INPUT type="text"
name="txtrammemory" id="txtrammemory"
runat="server"></td>
</tr>
<tr>
        <TD>Disco Duro</TD>
        <td><INPUT type="text"
name="txthardDisk" id="txthardDisk"
runat="server"></td>
</tr>
<tr>
        <TD>Nombre del Host</TD>
        <td><INPUT type="text"
name="txthostname" id="txthostname"
runat="server"></td>
</tr>
</table>
<div><INPUT type="submit" name="bsubmit"
value="Enviar">
<INPUT type="reset" value="Limpiar"></div>
</FORM>
</body>
</html>
```

la diferencia de un control tradicional de HTML y un HTMLControls, es agregando forzosamente a la etiqueta de apertura de formulario el atributo runat=”server”

```
<FORM name="forma1" method="POST"
runat="server">
```

y a cada uno de los controles controles los atributos id=”(nombre del control)” y...

runat="server", por ejemplo un control de texto y un control dropdown respectivamente.

```
<INPUT type="text" name="serialnumber"
id="serialnumber" runat="server">
<SELECT name="brand" id="brands"
runat="server">
```

Ahora estos controles deben de estar presentes en el código de la clase controladora, para poder interactuar con el formulario, así quedaría el código completo con los métodos de conexión, llenado de catálogos listo e inserción en la tabla principal.

```
using System;
using System.Web;
using System.Web.UI;
using System.Web.UI.HtmlControls;
using System.Web.UI.WebControls;
using System.Text;
using System.Collections;
using Npgsql;

namespace computo.classes{
        public class registro :
System.Web.UI.Page{
                //aqui van los controles de
la vista con HTMLControls
                protected
System.Web.UI.HtmlControls.HtmlInputText
serialnumber;
                protected
System.Web.UI.HtmlControls.HtmlSelect brands;
                protected
System.Web.UI.HtmlControls.HtmlSelect
operatingSystem;
                protected
System.Web.UI.HtmlControls.HtmlInputText
 rammemory;
                protected
System.Web.UI.HtmlControls.HtmlInputText
 hardDisk;
                protected
System.Web.UI.HtmlControls.HtmlInputText
 hostname;
                protected
System.Web.UI.HtmlControls.HtmlInputButton
Submit1;
                //aqui van las variables de
la conexion
                private NpgsqlConnection
dbcon = null;
                private NpgsqlCommand dbcmd
= null;
      private void Page_Load(object
sender,System.EventArgs e){
       //aqui para poder obtener los
catalogos
      foreach(ListItem item in
rListColeccion("Select id,name From
brands")) {
                    brands.Items.Add(item);
      }
      foreach(ListItem item in
rListColeccion("Select id,name From
OperatingSystems")) {
      operatingSystem.Items.Add(item);
           }

      }

      override protected void
OnInit(EventArgs e){
      this.Load += new
System.EventHandler(this.Page_Load);
      this.Submit1.ServerClick += new
System.EventHandler(this.Submit1_ServerClick
);
        }

        private void
Submit1_ServerClick(object sender,
System.EventArgs e){
         //va la consulta de insercción SQL
        string sql = String.Format("INSERT
INTO
Equipment(serialnumber,brand_id,operatingsys
tem_id,rammemory,hardDisk,hostname)VALUES('{
0}',{1},{2},'{3}','{4}','{5}')",serialnumber
.Value,brands.Items[brands.SelectedIndex].Va
lue,operatingSystem.Items[operatingSystem.Se
lectedIndex].Value,rammemory.Value,hardDisk.
Value,hostname.Value);
            //para debug only
            //Response.Write(sql);
            try{
            //aqui realizamos el insert
            conectar();
            dbcmd.CommandText = sql;
            Response.Write("(" +
dbcmd.ExecuteNonQuery().ToString() + ")
registros Afectados" );
           }catch(NpgsqlException ne){
           Response.Write(ne.ToString());
        }
  }

private void conectar(){
     try{
          dbcon = new
NpgsqlConnection(@"Server=127.0.0.1;Port=543
2;Database=inventario;User
ID=postgres;Password=chikoaze");
```

```
    dbcon.Open();
    dbcmd = dbcon.CreateCommand();
 }

catch(NpgsqlException e){
      Console.WriteLine(e.ToString());
      cerrar();
      }
}

private void cerrar(){
      dbcmd.Dispose();
      dbcmd = null;
      dbcon.Close();
      dbcon = null;
      Console.WriteLine("Cerrado...");
      }
      //funcion para llenar los HTMLSelect
      public ListItemCollection
rListColeccion(string sql){
      ListItemCollection myList = new
ListItemCollection();
      conectar();
      dbcmd.CommandText = sql;
      try{
         NpgsqlDataReader dr =
dbcmd.ExecuteReader();
         while(dr.Read()){
         for(int i = 1;i <
dr.FieldCount;i++){
         myList.Add(new
ListItem(dr[i].ToString().Trim(),dr[0].ToStri
ng().Trim()));
        }
      }
      dr.Close();
      cerrar();
      }
      catch(NpgsqlException e){
          myList.Add(new
ListItem(e.ToString().Trim(),"NULL"));
          cerrar();
      }
      return myList;
    }
  }
}
```

Primer paso: Métodos de apertura y cierre de la base de datos

Es muy importante tener el servidor de PostgreSQL en marcha, para que estos métodos funcionen y no lancen una excepción.

```
   private void conectar(){
       try{
           dbcon = new
NpgsqlConnection(@"Server=127.0.0.1;Port=543
2;Database=inventario;User
ID=postgres;Password=chikoaze");
           dbcon.Open();
           dbcmd = dbcon.CreateCommand();
           }
           catch(NpgsqlException e){
              Console.WriteLine(e.ToString()
);
              cerrar();
          }
       }

      private void cerrar(){
      dbcmd.Dispose();
      dbcmd = null;
      dbcon.Close();
      dbcon = null;
      Console.WriteLine("Cerrado...");
  }
```

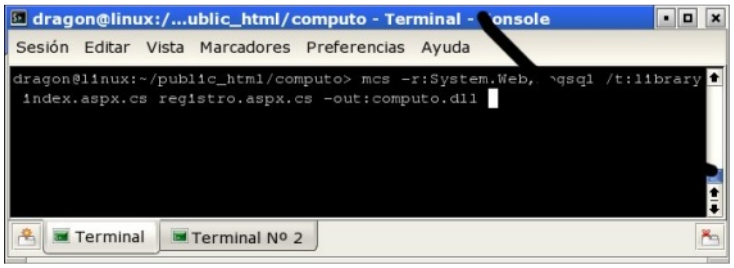
Una vez conectados hacer uso de los catálogos con el método siguiente:

```
public ListItemCollection
rListColeccion(string sql){
     ListItemCollection myList = new
ListItemCollection();
     conectar();
     dbcmd.CommandText = sql;
     try{
         NpgsqlDataReader dr =
dbcmd.ExecuteReader();
         while(dr.Read()){
            for(int i = 1;i <
dr.FieldCount;i++){
                myList.Add(new
ListItem(dr[i].ToString().Trim(),dr[0].ToStr
ing().Trim()));
            }
        }
    dr.Close();
    cerrar();
    }
    catch(NpgsqlException e){
        myList.Add(new
ListItem(e.ToString().Trim(),"NULL"));
        cerrar();
   }
   return myList;
}
```

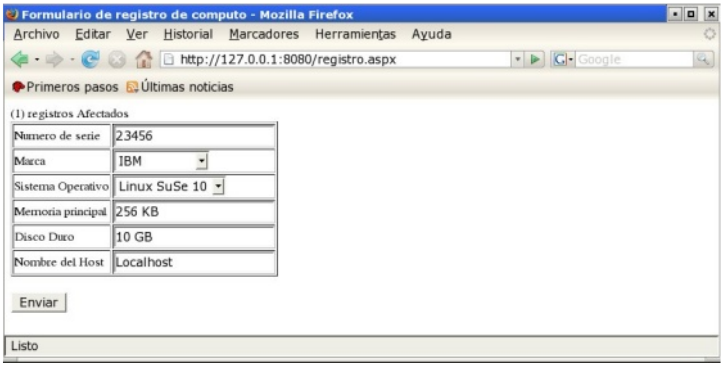
y por supuesto la creación del enunciado SQL para insertar, el método para poder insertar los datos:

```
string sql = String.Format("INSERT INTO
Equipment(serialnumber,brand_id,operatingsyst
em_id,rammemory,hardDisk,hostname)VALUES('{0}
',{1},{2},'{3}','{4}','{5}')",serialnumber.Va
lue,brands.Items[brands.SelectedIndex].Value,
operatingSystem.Items[operatingSystem.Selecte
dIndex].Value,rammemory.Value,hardDisk.Value,
hostname.Value);
try{
      conectar();
      dbcmd.CommandText = sql;
      Response.Write("(" +
dbcmd.ExecuteNonQuery().ToString() + ")
registros Afectados" );
      }catch(NpgsqlException ne){
           Response.Write(ne.ToString());
        }
```

Solo resta compilar:



y finalmente probar el formulario.



En la siguiente entrega se mostrará como hacer la página para visualizar los datos almacenados.

Si buscas referencias en linea, visita estos enlaces:

http://www.go-mono.com
http://dotgnu.org/

Proyecto Tor

comunicación anónima en la red

Andres Vargas **zodman@gmail.com**

¿Qué es TOR?

Tor como dice en su página es:

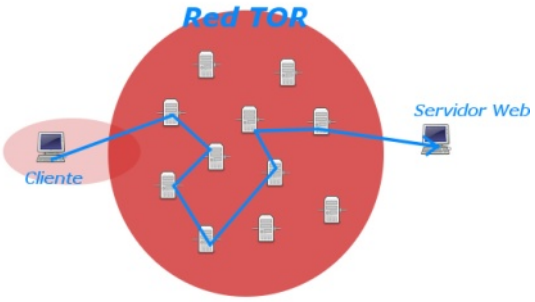
Es un conjunto de herramientas para un amplio abanico de organizaciones y personas que quieren mejorar su seguridad en Internet. Usar Tor puede ayudarte haciendo anónima la navegación y publicación Web, mensajería instantánea, IRC, SSH y demás aplicaciones que usan el protocolo TCP. Tor también proporciona una plataforma sobre la cual los desarrolladores de programas pueden construir nuevas aplicaciones que incorporen características de anonimato, seguridad y privacidad.

Básicamente lo que es Tor, es una gran red donde la red se dedica a descargar la información para ti. Como lo conocemos todos una gran red proxy.

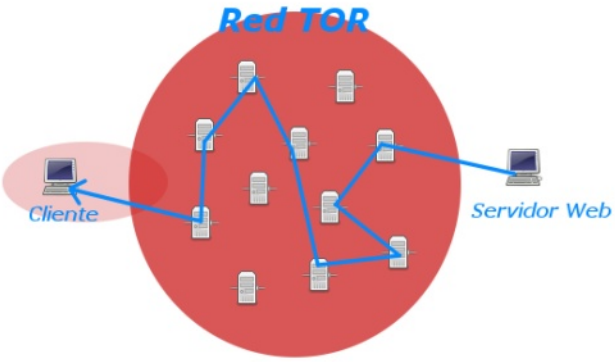
¿Como funciona TOR?

Tor es una implementación de una técnica llamada enrutamiento cebolla (por eso su logo es una cebolla), su concepto es transportar la información a través de una serie de cebollitas (equipos que funcionan como routers) y además de manera encriptada

Por lo que tenemos una red de ordenadores cada ellos enlazados a través de TOR. Cuando tú te unes a la red haces una petición de información, digamos que quieres abrir una página en un servidor Web.



Haces la petición a través de la red. Ésta enlaza nodos con otros nodos (se genera un camino de nodos), hasta llegar al servidor Web, hace la petición de información, y se regresa los datos a través de la red pero por un camino distinto al que se siguió, para hacer la petición.



En dado caso que vuelvas a hacer la petición de información al mismo lugar... pasa la petición hasta el servidor que quieres llegar pero con otro camino diferente.

Como resultado de todo esto, tu IP nunca fue usada para hacer la petición de información directamente hacia el servidor Web

por lo que otra IP fue usada.

Y es así como funciona Tor.

Hacerlo funcionar

Para empezar debemos bajar Tor, viene empaquetado para Debian, Fedora, Gentoo, Windows, MacOS y fuente.

Lo que yo hice fue bajar el archivo fuente para Linux/BSD.

Descompactarlo y compilarlo:

```
$tar xzf tor-0.1.1.26.tar.gz;
$cd tor-0.1.1.26
$./configure
$make
#make install
```

Tuve algunos problemas con los archivos, ya que necesita librerías de OpenSSL y libevent para que se compile

Pues bien al hacer make install se instala todo el árbol de archivos en /usr/local, por default la configuración de Tor, corre en modo cliente por lo que no hay que modificar nada en las configuraciones.

```
andy@lagrutadelzod:~$ tor
Dec 16 01:15:31.567 [notice] Tor v0.1.1.25.
This is experimental software. Do not rely
on it for strong anonymity.
Dec 16 01:15:31.682 [notice] Initialized
libevent version 1.1a using method epoll.
Good.
Dec 16 01:15:31.683 [notice]
connection_create_listener(): Opening Socks
listener on 127.0.0.1:9050
```

Ósea que Tor esta corriendo a la escucha por el puerto 9050. Que seria nuestro puerto de proxy. Y las aplicaciones se conectarían a este puerto para entrar a la red Tor.

Algunas herramientas:

Privoxy: http://www.privoxy.org/

Es un manejador/filtro de todo tus datos que pasan por la navegación Web. Este se puede configurar con el Tor para que funcione con esto, filtrando muchas cosas desde tus cookies, passwords del navegador, encabezados, adsens popus y demás cosas.

Algo muy importante es correr Tor con privoxy si lo que quieres es aprovechar el potencial del anonimato de Tor, ya que a veces el navegador hace las resoluciones de DNS, fuera de la red Tor, y las agrega a los headers de la petición (digamos http-headers) entonces con privoxy, filtra esta información o la cambia. Conservando tu anonimato.

Tor Botton del Firefox: https://addons.mozilla.org/firefox/2275/

Este botón es una extensión del Firefox, que puedes configurar para que el navegador use los puertos de Tor después de darle click, y evitar el molestoso cambio de configuración de proxy.

Más información en la documentación en español: http://tor.eff.org/docs/tor-doc-unix.html

Bien lo que sigue es instalar el Tor botton, en el navegador y configurarlo para que use Tor:

La prueba de fuego

Un método para ver si esta funcionando Tor es visitar alguna página que muestre nuestra IP en el navegador

http://www.my-ip.info/
http://whatsmyip.net

Y Verificar que no es la IP que nos da nuestro servicio ISP

O visitar el Tor tester

http://lefkada.eecs.harvard.edu/cgi-bin/ipaddr.pl?tor=1

Que nos dirá a través de que nodo estamos saliendo.

También visitar Google y darnos cuenta que no es nuestro Google de nuestro país.

Así es tienes tu trasero bien guardado!!

¿Torify IRC?

Puedes hacerlo!, usando las configuraciones proxy de tu cliente IRC, iguales a la del tor-botton, conectándose a localhost a través del puerto 9050

¿Torify SHH?

```
$ torify ssh  <parámetros>
```

Servicios Ocultos de TOR

Los servicios ocultos son servicios (Web, SSH, FTP, IRC) que tienen los servidores Tor y solo pueden ser alcanzados por clientes y servidores de Tor. Haciendo referencia por un nombre bien raro (encriptado) terminado en .onion

Tor tiene un wiki-hidden, si ya corres Tor, visita http://6sxoyfb3h2nvok2d.onion/tor/

Este wiki contiene tips, información de como sacar provecho de Tor, configurar software para Tor, etc.

Distribución BSD que trae Tor instalada: Anonym.OS

Es un CDLive, basado en OpenBSD, que fue desarrollada por un grupo de expertos en seguridad http://kaos.to/cms/content/view/14/32/

Ventajas:

Puedes usar Tor, para hacer descargas de servicio de hosting de archivos.

Desventaja:

Creo que la mas grande desventaja de Tor es su velocidad, ya como lo servidores de Tor, son donados, no brindan gran ancho de banda para ello. Por lo que esta por nosotros es colaborar al proyecto no solo siendo clientes, si no siendo servidores.

Para ser server en realidad estas donando ancho de banda; 20 kilobytes/s de subida y bajada (upload, download). Pero esto puede ser configurable.

Esto ha sido todo. 

Puedes obtener mas información en la página web del proyecto:

http://tor.eff.org/index.html.es

Conferencia sobre el proyecto Tor:

http://freehaven.net/~arma/wth1.pdf

Video (torrent) de la presentación de Tor en What The Hack?:

http://rehash.waag.org/WTH/wth-anonymous-communication-58.mp4.torrent

FreeBSD y Bluetooh
manejando dispositivos

Lael Gónzalez Rosales **optix@zonartm.org**

Me hubiese gustado escribir para esta publicación un texto más profundo y de igual modo interesante, no queriendo decir de que este no lo sea, pero la falta de tiempo y los cambios que se suscitan hacen que no sean las mismas oportunidades de antes, para dedicarle a los sistemas.

Pienso que a muchos les ha de interesar emparejar sus dispositivos bluetooth bajo este estupendo sistema (FreeBSD) y jugar un poco con el, así que describiré la forma de como hacerlo y algunas cosas que se pueden lograr.

¿Que pretende este articulo?

Bien, mas que todo demostrar que FreeBSD ofrece soporte a nuevas tecnologías que cada ves emergen con mas frecuencia. Igualmente ser una alternativa a otros sistemas como Microsoft Windows o GNU/Linux en sus diferentes sabores con igual o mejores experiencias.

Como pueden ver este texto no esta dirigido a usuarios expertos o gurus, ya que en realidad todo lo que realizaremos son procesos sencillos pero que debes tener cuidado para no cometer errores.

Podemos mencionar brevemente sobre bluetooth como una tecnología para redes de ámbito personal con un alcance de 10m, construida en modo ad-hoc. Crea un perfil para actuar como servidor de ficheros basado en FTP, transporte de voz, emulación de línea serie, etc. Para mas información o si en realidad no tienes ni la mas remota idea de lo que estamos hablando, pillate en la red muchos artículos que hablan muy a fondo sobre el tema.

Empezando

La pila bluetooth en FreeBSD trabaja bajo el entorno de netgraph que provee funciones para conexiones, networking al kernel entre nodos e implementar diferentes protocolos.

Los drives de soporte son dados al kernel por ng_ubt basado en la especificación v1.1 de bluetooth sobre dispositivos USB.

Reconociendo el Dispositivo conectado:

El primer paso será cargar al kernel el módulo ng_ubt para ello solo pondremos:

```
# kldload ng_ubt
```

Conectamos nuestro dispositivo usb bluetooth y reiniciamos el sistema para mirar si carga el nuevo modulo que hemos solicitado:

```
# shutdown -r now
```

Al iniciar nuestro sistema añadira al loader.conf la opcion:

```
ng_ubt_load="YES"
```

si todo es correcto nos podemos fijar que en el arranque y la carga de los dispositivos encontrados nos mostrara el siguiente mensaje entre otros.

Si no paso nada, podemos hacerlo manualmente y no debería de fallar. vamos al directorio:

```
/boot/defaults/loader.conf
```

Con algún editor de texto abrimos el loader.conf y buscamos los Módulos de Netgraph, que suelen estar antes de Sound Modules y hubicamos la linea ng_ubt_load="NO" y reemplazamos el NO por SI, en caso de no estar la pondremos de tal modo que quedaría así:

```
ng_ubt_load="YES"
```

Lo guardamos y salimos, todo debería de ir como la seda =).

Para activar la pila del bluetooth usaremos un script de ejemplo que esta en /usr/share/examples/netgraph/bluetooth/rc.bluetooth. Este nos servirá para activar el dispositivo y que este listo para estar visible por otros componentes con bluetooth.

Para su correcto funcionamiento copiaremos el script a /etc/rc.bluetooth nos resta ponerlo en marcha:

```
# /etc/rc.bluetooth start ubt0
```

Veremos el siguiente mensaje:

(En caso de que te salga el mensaje "permiso denegado", cambia los permisos del archivo a 770.)

Podemos activar la búsqueda de nuevos dispositivos desde el celular y nos saldrá algo

Como:

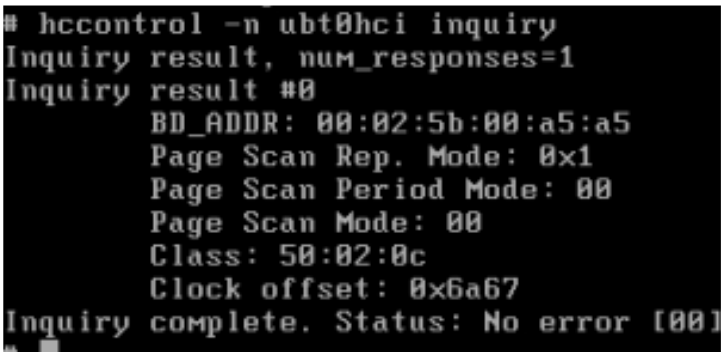
host.(ubt0) <--- Nuestro dispositivo USB en FreeBSD.

Modo de operación (interfase y control de comunicación)

La comunicación se realiza mediante un HCI (Host Controler Interface). Facilitando una capa de acceso igual para todos los dispositivos bluetooth, esta capa interactúa directamente con el firmware del dispositivo bluetooth y permite acceder al estado del hardware como a los registros de control.

Los dos nodos se crean mediante Netgraph, tanto en el dispositivo (Netgraph L2CAP), como en la maquina (Netgraph HCI).

La interfaz de comandos se basa en una utilidad llamada hccontrol, y es la que nos servirá para algunas operaciones básicas a la hora de trabajar. Buscando dispositivos en el rango



Los dispositivos bluetooth a manera similar que una tarjeta de red Ethernet, manejan una dirección que la identifica frente a las demas, es la que observamos como BD_ADDR, por consiguiente si queremos saber el nombre de ese dispositivo que en mi caso seria 00:02:5b:00:a5:a5 usamos el comando Remote_Name_Request

```
# hccontrol -n ubt0hci remote_name_request
00:02:5b:00:a5:a5
BD_ADDR: 00:02:5b:00:a5:a5
Name: OpTix
```

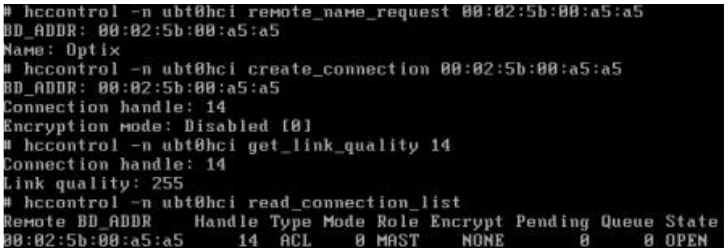
Bien, es momento de realizar una conexión con nuestro dispositivo bluetooth que tenemos, sea un celular u otro dispositivo inteligente.

```
# hccontrol -n ubt0hci create_connection
00:02:5b:00:a5:a5
BD_ADDR: 00:02:5b:00:a5:a5
Connection handle: 14
Encryption mode: Disabled [0]
```

Hay que mencionar que al hacer esto debemos vincular los dos dispositivos y tener habilitadas las opciones de visibilidad, podemos leer la conexión para ver que en realidad existe:

```
# hccontrol -n ubt0hci read_connection_list

Remote BD_ADDR       Handle Type Mode Role
 Encrypt Pending Queue State
00:02:5b:00:a5:a5     14   ACL    0   MAST
   NONE        0     0  OPEN
```



Mirando la calidad de la señal:

```
# hccontrol -n ubt0hci get_link_quality 14
Connection handle: 14
Link quality: 255
```

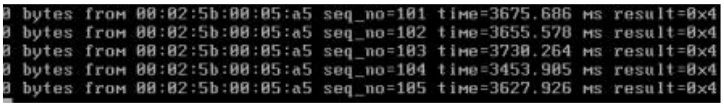
Esto nos dice que el enlace esta a su mejor nivel de señal (255) por ultimo hacer la desconexión:

```
# hccontrol -n ubt0hci disconnect 14
conection handle: 14
Reason: Connection terminated by local host
[0x16]
```

El handle que manejemos en cada conexión es muy importante para aplicar los comandos pues es la identificación del entable entre los dispositivos. Si deseas conocer otros comandos disponibles y usos visita las paginas de documentación de cada comando.

¿Algo no va bien ?

Es posible que al momento de abrir una conexión se desconecte después de algunos segundos, en el protocolo L2CAP (Logical Link Control and Adaptation Protocol) que es el que proporciona los servicios de datos a conexiones. Esto es debido a que no se están manejando datos o paquetes y se cierra el nodo que trabaja a modo de “sockets” dando como resultado =0x4.



Podemos modificar algunas opciones del ng_l2cap pero nos llevaría mas tiempo.

Soluciones y Transferencia de archivos

Puedes Activar las conexiones de modo permanente del siguiente modo:
Después de hacer un inquiry sobre que dispositivos están disponibles vamos a

/etc/bluetooth/hosts y lo editamos ingresando los datos de nuestro celular.

```
00:02:5b:00:a5:a5 optix
```

Recuerda que debes de quitar el signo que comenta la linea #, salir y guardar cambios.

Activando conexión

```
# l2ping -a optix
0 bytes from optix seq_no=0 time=48.633 ms
result=0
0 bytes from optix seq_no=1 time=37.551 ms
result=0
0 bytes from optix seq_no=2 time=28.324 ms
result=0
0 bytes from optix seq_no=3 time=46.150 ms
result=0
.....
.......
```

Con esto mantenemos activos los enlaces y podemos darnos cuenta porque se ha activado el servicio de bluetooth en nuestro celular, reportando conexión.

Transferencia de archivos

OBEX Push

OBEX es un protocolo para la transferencia de archivos entre dispositivos, los usos mas comunes es sobre puertos infrarrojos.

```
# cd /usr/ports/comms/obexapp
# make install clean
```

Esta herramienta tiene soporte para varios comandos tipo FTP, por mencionar algunos: CD, delete, disconnect, empty, get, ls, put, mkdir, etc. Un ejemplo seria:

```
# obexapp -a optix -C OPUSH
obex> get
get: remote file> prueba.txt
get: local file> prueba.txt
Success, response: OK, Success (0x20)
```

Para el servicio de OBEX es necesario que este corriendo el servidor sdpd, los objetos por lo general se guardan en /var/spool/obex.

Truco: en los nokia, presionando *#2820# nos saldrá la dirección BD_ADDR del cel.

Desconozco si en modelos de otras compañías sea igual.

Bien, con esto doy por terminado el tema, ya que hemos cumplido con el objetivo. En próximas oportunidades estaremos hablando de otras utilidades que son posibles realizar mediante bluetooth y que atañe a un tema de redes avanzadas, así mismo explicar un poco los protocolos usados para estos funcionamientos, etc.

Para obtener mas información o resolver tu dudas visita:

http://www.zonartm.org

Diseccionando la manzana

MacOS X mas que la niña bonita

Julio Acuña Carrillo **julio.acuna@revista-sl.org**

Un poco de historia.

Cuando Steve Jobs fue expulsado de Apple a mediados de los 80's fundó NeXT, una compañía que vendía hardware con un sistema operativo tipo UNIX llamado NeXTSTEP, lo que diferenciaba este sistema de otros sistemas operativos de la época era la suma de todos sus componentes de los que hablaremos mas adelante.

Después de una época no muy buena para Apple, la compañía deseaba mejorar su sistema operativo, lo cual no logra realizar y decide incorporar un nuevo sistema operativo. Apple tenía varias opciones, utilizar BeOS, NeXTSTEP o Windows (recordemos que Apple es una empresa de hardware).

Aunque BeOS había ganado cierta reputación como un muy bueno y moderno sistema operativo, Apple decidió no ir por el plan B como ellos le decían y se inclinaron por NeXTSTEP. Hacia 1997 NeXT fue comprada por Apple y eso trajo a Steve Jobs de regreso a la compañía que cofundara con Steve Wozniak y el nacimiento del OS X.

En el fondo las Machintosh actuales son descendientes de las NeXTcube[1] o las NeXTstations, los procesadores que utilizaban las NeXT eran de la familia Motorola 68k, el mismo utilizado en las primeras Machintosh.

Al final de sus días NeXT dejó de producir hardware y se dedicó enteramente al software, portando su sistema operativo a las arquitecturas PowerPC, SPARC e Intel.

Un sistema desde las tripas.

Actualmente mucha gente queda fascinada por la interfaz gráfica del Mac OS X al extremo de querer imitar su look and feel modificando el escritorio, instalando temas, iconos y desklets que asemejan al sistema de la manzana, pero dejan de lado varias tecnologías que podemos tener en sistemas operativos libres de la actualidad.

El kernel.

OS X utiliza un kernel híbrido, por un lado tenemos el microkernel mach, el mismo que se puede encontrar en HURD, junto con componentes de FreeBSD (anteriormente 4.3BSD) y un API para escribir controladores llamado I/O Kit, todo esto es lo que se conoce como XNU. En NeXTSTEP se favorecía la programación orientada a objetos, es por eso que frecuentemente encontramos la palabra Kit.

La teoría de la evolución.

Encima del kernel XNU está lo que se conoce como Darwin, un sistema operativo con licencia libre[2] pero controlado por Apple. Si quisiéramos tener un sistema funcional podríamos utilizar Darwin como punto de partida, existen proyectos basadas en Darwin como OpenDarwin o GNU-Darwin que permiten construir un sistema operativo completo con aplicaciones 100% libres. En corto, Darwin es lo que queda si a OS X le quitáramos la interfaz gráfica, las aplicaciones propietarias y algunos controladores propietarios; eso nos dejaría un OS X irreconocible. :)

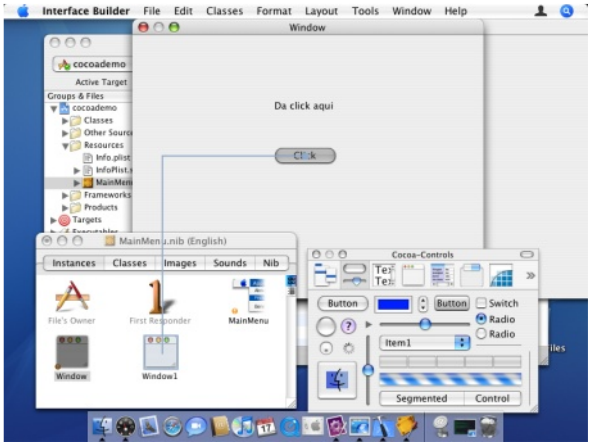
El siguiente paso.

En Sun Microsystems estaban interesados en correr el ambiente NeXTSTEP --que era el mas avanzado en su época-- en sus sistemas Solaris así que en alianza con NeXT crearon la especificación OpenStep, una plataforma separada del sistema operativo que consistía básicamente en el Display Postscript System (DPS), compilador, runtime y bibliotecas Objective-C[3]. Actualmente OS X utiliza Quartz como sistema gráfico en sustitución de DPS y Cocoa que es una implementación propietaria de OpenStep.

Una implementación libre de OpenStep es GNUstep que inicialmente fue desarrollada por Paul Kunz al tratar de portar una aplicación de NeXTSTEP a otras plataformas pero al final terminó por reescribir la biblioteca en la que dependía dicha aplicación, una vez que la especificación de OpenStep se hizo pública este proyecto creció hasta ser lo que hoy conocemos como GNUstep.

GNUstep es mas o menos compatible con Cocoa, así que si se escribe una aplicación en GNUstep seguramente correrá en Cocoa, pero no es seguro que una aplicación escrita en Cocoa corra en GNUstep. Aunque GNUstep fue hecho con UNIX en mente, se puede instalar en Windows y correr las mismas aplicaciones.

Para handhelds o dispositivos móviles podemos usar QuantumSTEP, un derivado de GNUStep para la Sharp Zaurus y otros dispositivos con Linux instalado.



Construyendo una aplicación en Cocoa

Construyendo y enlazando.

A veces vemos aplicaciones para OS X que se crean muy rápido, esta es la ventaja de las plataformas basadas en OpenStep; como todo es un objeto podemos construir nuestra aplicación basándonos en objetos previamente hechos, enlazándolos y añadiendo pocas líneas de código; ¿a qué me refiero con esto?, para construir una aplicación con interfaz gráfica solamente debemos añadir los objetos de los que está compuesta dicha aplicación haciendo un simple drag and drop y con el mismo mouse interconectar los objetos para que realicen una tarea específica (ver figura #1 Cocoa), no nos tenemos que preocupar en cómo lo hacen, los objetos entienden el mensaje que se les envía y a su vez mandan un mensaje a otros objetos si es necesario. Simplemente con utilizar los Kits adecuados en poco tiempo tendremos una aplicación funcionando y en producción; lo que tomaría meses nos toma una semana en hacerlo de esta forma.

También podemos construir nuestros propios objetos e incorporarlos en nuestras aplicaciones de la misma forma que lo hacemos con los objetos que ya tenemos disponibles en los Kits.

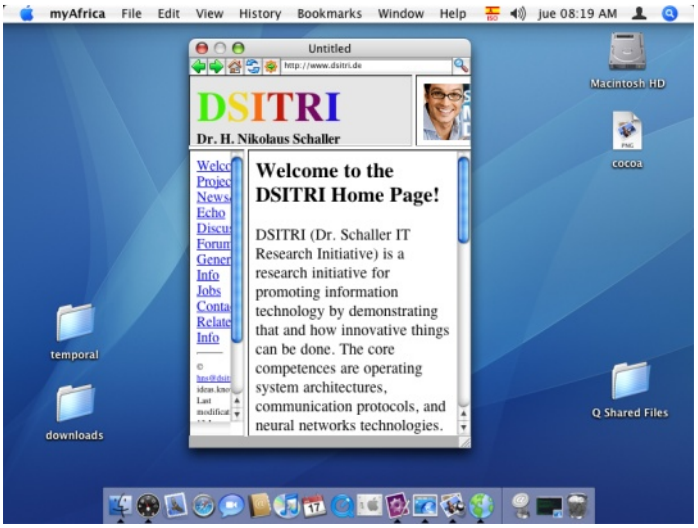
Las aplicaciones construidas de esta forma nos dan un .app que es un directorio donde se encuentran tanto el binario como algunos otros directorios y archivos que utilizan las aplicaciones como son los iconos y archivos xml.

En la Mac tenemos Xcode e Interface Builder para construir aplicaciones en Cocoa, en GNUstep tenemos GCC[4], Gorm y ProjectCenter que serían los equivalentes.

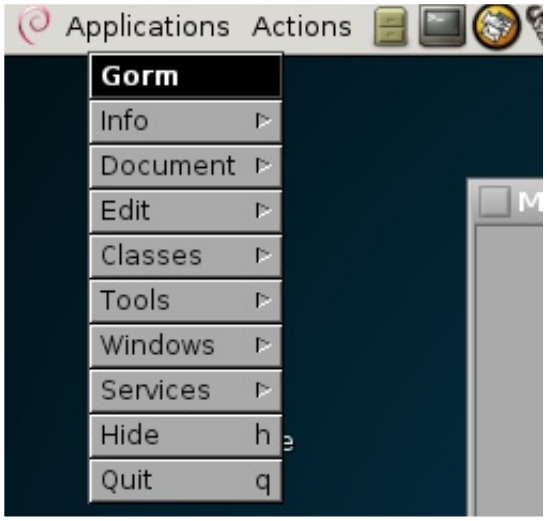
Otras cosillas.

Los componentes de los que hablaremos a continuación no son menos importantes en Mac OS X pero no me quiero extender mucho sobre ellos.

Finder: El Finder es la aplicación gráfica que todo mac-ero conoce muy bien, es tanto un manejador de archivos como una shell...



MyAfrica, una aplicación de QuantumSTEP en la Mac



GNUStep: se nota el parecido entre frameworks

gráficas basada en las versiones anteriores del Mac OS y reescrita para la versión X.

Carbon: Carbon es un conjunto de APIs con los cuales se pueden portar aplicaciones de Mac OS Classic (versión 9 y anteriores) a OS X, así fue como Apple facilitó la transición de una versión a otra del sistema para los usuarios.

Aqua: El encanto de OS X se lo debe a Aqua que es el sistema de ventanas que tiene una apariencia que han tratado de imitar, como cierto sistema operativo propietario de reciente aparición.

Exposé: Aunque la Machintosh siempre ha sido pensada para que sus usuarios tengan una mayor facilidad de uso del sistema operativo, uno tiende a perder alguna ventana si se tienen muchas aplicaciones abiertas, es por eso que Exposé nos brinda la posibilidad de encontrar la ventana deseada mostrando todas las ventanas abiertas en vistas miniatura sobre el escritorio. Una aplicación similar en el mundo del software libre se llama Skippy sólo que éste corre sobre las X.

Dock: El Dock del OS X ha entrado en varias polémicas sobre usabilidad pero es cierto que a muchos les gusta cuando los iconos aumentan de tamaño al paso del cursor del ratón sobre el Dock. Exposé tiende a complementar muy bien las carencias del Dock y si, adivinaron, el Dock es un legado de NeXTSTEP.

Universal Binary: La transición de procesadores PowerPC a Intel en las nuevas Mac no hubiera sido posible si no existiera una forma de hacer que las aplicaciones fueran multiplataforma, lo que Apple llama Universal Binaries (antes fat binaries) son aplicaciones compiladas para las dos arquitecturas sobre las que corre OS X en un mismo archivo.

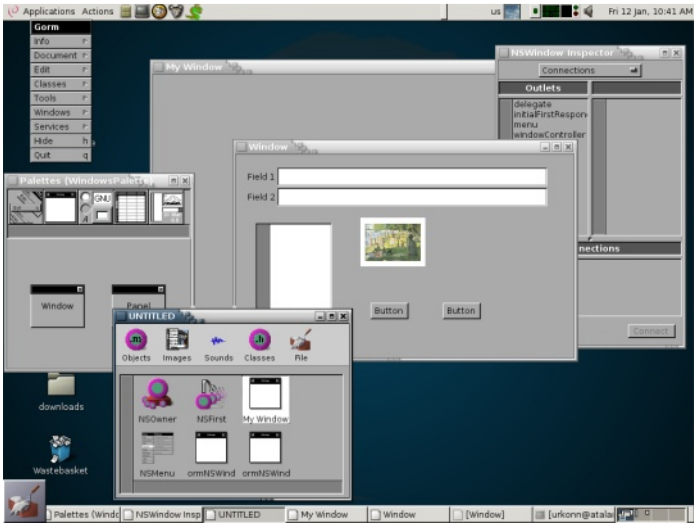
Mach-O: OS X es tal vez el único sistema operativo que utiliza Mach-O en lugar de otro formato de archivos binarios como ELF.

DMG: Las aplicaciones que se distribuyen para la Mac están empaquetadas en una imagen de disco dmg.

Cuando nos encontramos con iconos o wallpapers en dmg no tenemos que envidiar a los usuarios de Mac, podemos montar esa imagen con el siguiente comando y utilizarlos en nuestros escritorios.

```
mount -o loop -t hfs  iconos.dmg
 /punto/de/montaje/
```

Como está basado en Darwin, en OS X podemos instalar aplicaciones libres con herramientas como Fink o Darwinports, sin embargo para algunas aplicaciones gráficas necesitaremos un servidor X11 corriendo.



Gorm corriendo sobre Gnome

¿GNUMaker?

Muchos tienden a confundir GNUstep con WindowMaker, el primero es un framework y el segundo es un manejador de ventanas que imita el look del NexTSTEP original, además de ser un entorno bastante ligero y agradable para trabajar.

Como podemos ver, el OS X es mas que una cara bonita, sin embargo muchas de las tecnologías que emplea las podemos usar en nuestros Sistemas Operativos Libres.

(1)Fue en un NeXTcube donde se crearon el primer navegador y el primer servidor web por Tim Berners-Lee.
(2)La licencia Apple Public Source Licence es libre pero incompatible con la GPL.
(3) Objective-C es un lenguaje orientado a objetos que se puede ver como una capa encima del lenguaje C con un poco de la sintaxis de Smalltalk.
(4)Hace tiempo que GCC soporta Objective-C

FreeSBIE Live CD
free system burned in economy

Hector Leal Morales **hector.leal@revista-sl.org**

Hace poco el grupo de edición de la RevistaSL votó que el siguiente numero a publicar seria acerca de BSD e invito a muchas personas a participar en este nuevo número. Entonces como no se mucho del tema de BSD decidí buscar una manera fácil de comenzar con esto.

Recuerdo que en el año 2004 en un taller impartido por Interlegit.com.mx de nombre Linux Distribuciones, al terminar el mismo ya después de la sesión de preguntas y respuestas, alguien me pregunto acerca de FreeBSD y pues no había tenido mucho contacto con BSD mas que una instalación en PC y solo pude recomendar algunas paginas de consulta y algunos proyectos que sabia que existían. Entre esos proyectos recuerdo alguno tipo LiveCD, así que me puse a buscar un poco para ver si aun existían y que había sucedido con ellos.

El proyecto que recuerdo es FreeSBIE, su URL es http://www.freesbie.org

Free System Burned In Economy (FreeSBIE) es un LiveCD basado en el sistema operativo FreeBSD, que bueno al ser un LiveCD basta con ponerlo en el lector de CD o DVD de una PC e iniciar la misma de forma que utilice el dispositivo de CD o de DVD para utilizar el sistema operativo cargado en el disco, sin instalación en el disco duro.

En el sitio de FreeSBIE actualmente esta disponible la imagen ISO así como torrent del FreeSBIE 2.0-RC1 y también la imagen del estable FreeSBIE 1.1

FreeSBIE cuenta con diferentes apartados informativos, creo que uno de los primeros que se debe de leer es la FAQ (www.freesbie.org/faq.html), y por supuesto revisar los screenshots que por el momento solo están de la versión estable. La documentación de FreeSBIE esta disponible en http://wiki.freesbie.org, ya que descarguen la imagen y creen el disco, pueden iniciar el equipo de computo.

Comienza la carga del sistema elijen la entrada por defecto y podrán observar lo siguiente, FreeSBIE 2.0RC1, Based en FreeBSD 6.2-RC1, Noviembre de 2006.

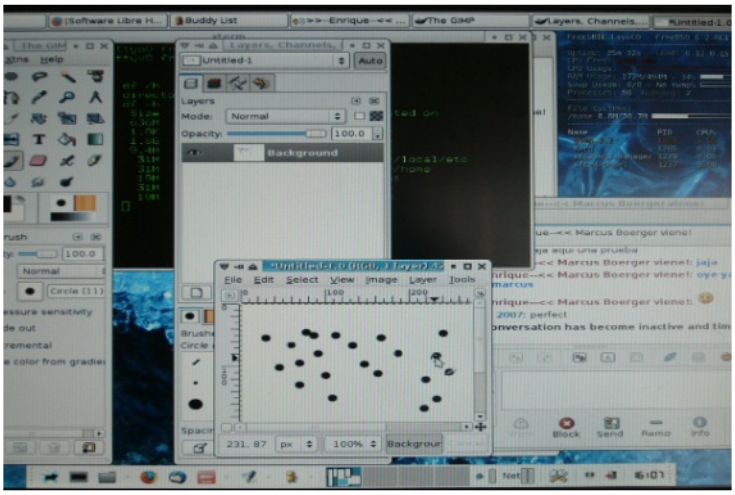
En el proceso se reconocen los dispositivos, el quemador, la red, se crean conexiones de red. Crea algunas particiones y comienza en la línea de comandos que esta después un dibujito en ASCII de un diablito.

El directorio personal es /usr/home/freesbie

Con un recorrido rápido por los directorios, se pueden ubicar algunos archivos de configuración y sistema, y cosas así, tal y como en Linux.

Después con el comando "df -h" se pueden ver las particiones que tiene el sistema y bueno para no seguir en la línea de comandos presionan starx para ver que hay con la parte gráfica.

Inicia de forma muy amigable y con un ratón que parece tener dolor de estomago.



Se pueden mover entre terminales con ALT + CTRL + (de la F1 a la F9 la F9 es la parte gráfica).

Al parecer se tiene Xfce con un menú en la parte de abajo en donde se pueden encontrar varias aplicaciones que son familiares como el AbiWord, Emacs, Gnumeric, VIM, Xpdf, aMule, Ettercap, Firefox, Gaim, Gftp, Nmap, Remotedesktop, Thunderbird, wireshark, Xchat, Mplayer, GQview, gThump, Inkscape, GIMP y SciTE entre otros.

Todo es bastante rápido y amigable la red funciona perfectamente, me conecto con GAIM saludo a Chalo, Enrique, Cloe, Maggit y M3nt3, recibo mensajes y con esto pruebo que el audio esta funcionando. Inicio Firefox, abro el sitio de la revista, también abro una Terminal y el GIMP y todo funcionando bien. El escritorio nos indica el uso del CPU y de memoria RAM, al momento de escribir estas lineas 20% de CPU y 34% de la memoria RAM.

Apago la máquina con el shutdown -h now y funciona.

Bueno definitivamente la experiencia es motivante al utilizar FreeSBIE. Esto me demuestra que no hay ningún motivo para no probar una instalación en la PC y aprender sobre BSD.

Para obtener mas información de este proyecto, visita estos enlaces:

http://www.freesbie.org
http://www.freebsd.org

Del **13** al **16** de
Febrero de 2007
Facultad de Ingeniería
Ciudad Universitaria
México, D.F.

CONSOL

CONGRESO NACIONAL DE SOFTWARE LIBRE 2007

"Convergencia Tecnológica del Siglo XXI"

Temas

- Comunidad, Filosofía y Negocios
- Desarrollo de Software
- Aplicaciones
- Administración, Seguridad y Redes
- Robótica y Hardware

www.**consol**.org.mx

GULEH

Grupo de Usuarios Linux del Estado de Hidalgo

Josue Gutierrez **josue.gutierrez@revista-sl.org**

¿Qué es el GULEH?

El Grupo de Usuarios de GNU/Linux del Estado de Hidalgo se encuentra conformado por un equipo de personas participativas y con intereses comunes, dedicadas a la difusión, uso y aprendizaje del sistema operativo GNU/Linux así como de proyectos y herramientas basadas en el Open Source.

Misión

Fomentar la filosofía del Software Libre con el fin de generar y compartir conocimiento, además de servir como espacio para la formación integral y profesional de sus integrantes.

Objetivos

Promover el uso y desarrollo de Software Libre y/o código abierto en el ámbito académico, científico y empresarial en el estado de Hidalgo, desarrollando soluciones de tecnologías de información sobre GNU/Linux.

Miembros.

La junta directiva esta integrada por las personas más activas del grupo, las cuales participan de manera física apoyando en actividades tales como:

- Coordinación de proyectos.
- Visitar áreas de promoción.
- Administrar el servidor del grupo.
- Además se cuenta con miembros virtuales quienes apoyan en la lista de usuarios resolviendo dudas e inquietudes de los inscritos.

Estructura jerárquica

El GULEH no es una democracia, es una meritocracia, basada en el compromiso y responsabilidad que se tiene en las actividades, no importando el nivel de conocimientos técnicos en la materia.

Para motivar la participación de los miembros y basándose en los méritos, se han designado tres grupos principales:

Iniciados: Es el novato o aquel que tiene ganas de compartir conocimiento y experiencia. Todos los interesados en colaborar con el grupo pasan por este nivel, es un punto en el que se pone a prueba la disposición y responsabilidad, para ganar el derecho a participar de forma activa.

Activistas:

Este miembro apoya de forma presencial o no, cuenta con el reconocimiento de los miembros fundadores y demás activistas (no

necesariamente todos). Ya tiene voz en la toma de decisiones con respecto al rumbo que puede tomar el grupo.

Fundadores: Son los miembros que han colaborado continuamente y que ocasionalmente sacrifican recursos personales para beneficio del grupo (tiempo, dinero, equipo). En conjunto toman las decisiones sobre las actividades que se desempeñaran en cuanto a la administración y difusión.

Visita la página oficial del GUL para mas información:

http://www.guleh.org

Suscribete a la lista de correos:

http://www.red-libre.org/mailman/listinfo/guleh

Si deseas visitar documentos de GULEH:

http://docs.guleh.org

Ademas puedes platicar con los miembros del GULEH en:

Canal #guleg
Red irc.freenode.net

GULEV 2006

el evento de software libre con mas tradición

José Luis Galicia jose.luis@revista-sl.org
Jesus Antonio Balam jesus.antonio@revista-sl.org

Del 7 al 9 de Diciembre. En el hotel Gran Meliá de la Cd. de Cancún, Q. Roo se llevó a cabo el GULEV Congreso Internacional de Software Libre 2006.

Estuvieron invitados de talla internacional como: Miguel de Icaza, Federico Mena, Rasmus Lerdorf, Guido Van Rossum, Bruce Momjian, Bdale Garbee, Xavi, Eduardo Sacristán, Patrick Vielle, entre otros.

La siguiente reseña es a partir de la perspectiva que tuvieron tres integrantes de la Comunidad Linux Chetumal y editores de RevistaSL en el congreso:

DIA 1

El 7 Dic a las 2 am, Antonio, Gonzalo (El editor en jefe) y su servidor (José), partimos rumbo a Cancún en auto (el "yaris" de José), nuestro destino era llegar al GULEV y pasarla muy bien con todos los amigos de la comunidad y por supuesto aprender mucho más del maravilloso mundo del Software Libre. Después de un viaje agotador por fin llegamos alrededor de las 7 am todos medios dormidos y cansados llegamos a Cancún.

Alrededor de las 11:00 arribamos al hotel, averiguamos dónde se debía hacer el registro, después de dar unas vueltas, subir y bajar unas cuantas escaleras (hicimos nuestra dotación de ejercicio) encontramos el lugar. Después de hacer el registro y pedir nuestro respectivo descuento de estudiante y trabajador educativo además de que pagamos a meses sin intereses, fuimos al lugar donde se llevarían a cabo las primeras conferencias- Como era de esperarse aparecieron los problemas propios de los congresos, las conferencias no se llevaron de acuerdo a como se tenían planeadas, se estuvo modificando el programa, y pues no hubo buena comunicación por parte de los organizadores aunque igual nosotros nos debimos haber acercado a preguntar que era lo que estaba pasando.

Ya cuando se había terminado la hora programada de conferencias nos fuimos a comer a un supermercado, Andábamos con la interrogante de ¿por qué no hubo conferencias?, ¿nos habríamos confundido?, ¿Qué habrá pasado? Todas estas preguntas fueron contestadas al regreso de la comida ya que resulta que el programa original sufrió un cambio y las conferencias que estaban programadas para la mañana se pasaron a la tarde y se hicieron otros ajustes en los siguientes dos días, la verdad no los recuerdo muy bien.

Llegamos al hotel como a las 3.30 y empezamos a saludar a los amigos que ya habían llegado, a eso de las 4:00 pm dio inicio la primera

conferencia, se trató de Kioscos informáticos, la cual fue impartida por un cuate de la UNAM llamado Eduardo Sacristán, para que se den una idea, un kiosco informático es una computadora que puede o no estar conectada a la red o Internet que le proporciona información a la gente sobre un lugar, servicios, etc., algo así como las terminales que tiene Telmex en sus oficinas y uno puede consultar su saldo, datos, teléfono, etc. Mediante un Live CD y Firefox, podemos cargar aplicaciones informáticas con pocos recursos computacionales. Mencionó ejemplos donde él ha utilizado dichos kioscos, como por ejemplo las elecciones presidenciales de México que acaban de pasar.

Al término de esta conferencia siguió otra denominada “Asegurando redes inalámbricas con WPA-PEAP usando FreeRadius” esta la dio Patrick Vielle, al comienzo estuvo divertido porque comenzó a hablar en términos médicos, se presentó como doctor además que comenzó presentando fotografías de unos estudios médicos, todos con nuestra cara de "what?" durante unos minutos, después el Dr. Patrick, título que se le quedó, mencionó que todo eso había pasado él por ser tan bilioso y ahora se tomaba la vida de una manera más divertida y con menos presiones. Ya que el congreso no esta destinado a un publico en general sino que abarca temas diversos, no todos pudieron entender la conferencia en su totalidad, ya que a algunos les llama mas la atención la programación, a otros les llama las redes, otros la seguridad en general, Por cierto durante esta conferencia hizo su aparición una persona de lo más extraña que estaba tome y tome fotos e ya hasta después de un buen rato nos percatamos de que se trataba de Miguel de Icaza que estaba presumiendo su cámara rebel.

Al final de las conferencias nos pusimos de acuerdo con los ponentes para ir a cenar, la cena fue muy amena y divertida, pudimos convivir con la mayoría de ellos y pues la verdad son personas muy sencillas, amables y dispuestas a pasar un buen rato de forma sana y divertida, compartiendo una buena comida, una cerveza, una platica interesante, hubiera sido interesante entender que tanto estaban platicando Miguel y Bdale Garbee, creo que este año debo empezar a practicar mi vocabulario.

Día 2

El segundo día José y Antonio estuvieron en el taller de PHP llamado PHP tips & tricks, éste duró de 10:00 am a 2:00 pm y fue impartido por ni mas ni menos que Rasmus Lerdorf, si te preguntas quién es él ?, la respuesta es: El creador de PHP, juaz, juaz, juaz, gracias a él, es como gano mis centavitos juar juar, No entramos a ninguna de las conferencias matutinas. El taller estuvo súper interesante, estuvo platicando sus experiencias en cuanto al desarrollo, como ha implementado nuevas tecnologías al lenguaje, estuvo dando varios tips y tricks acerca del lenguaje. Aquí solo un comentario, algunas personas como José y Antonio sintieron que se pudo haber aprovechado mas el curso si hubiera existido la posibilidad de hacer traducción simultanea, eso fue algo que le reclamamos a

a Miguel el organizador del GULEV pero nos dijo que éramos las terceras personas que preguntaban sobre la traducción, creo que si bien es nuestra obligación el prepararnos lo mejor posible también creo que seria deseable la traducción para un mejor aprovechamiento se pida o no. Otra cosa que es digno de mencionar es que se vio una buena participación del sector femenino en el congreso, tanto como ponentes como los que fueron a escuchar y tomar los talleres.

Por la tarde hubieron dos conferencias, a la primera no entré así que no se de que se trató, la segunda fue:"Pregúntale a Miguel", en ella Miguel de Icaza llegó y nos dijo a todos los que estábamos en el salón, pregúntenme de lo que quieran: Novell, política, religión, pueden preguntarme lo que se les ocurra. Hubieron algunas preguntas, como por ejemplo el acuerdo entre Novel y Microsoft, sobre Mono, Software libre, entre otros temas variados, la verdad muy interesante y se nota que Miguel está muy bien preparado, no sólo en la parte de negocios y programación, sino en cultura general. Tardó como 1 ó 2 horas, el salón estuvo abarrotado de gente.

Para finalizar ese día fuimos a fraternizar al mirador, con unas cuantas cajas de cerveza nochebuena, Sabritas, pizza y otras botanas variadas, estuvimos platicando, compartiendo experiencias y haciendo un breve análisis sobre como se había ido dando el congreso.

Día 3

Después de habernos desvelado de sobremanera, nos levantamos como pudimos para llegar a la conferencia de PostgreSQL con Bruce Momjian, Estuvo interesante el enfoque que Bruce le dio a la platica más que el aspecto técnico fue haciendo analogías con la historia y como se han ido los cambios en cuanto a comunicación, transporte y mas que nada la ideología de las personas. Por cierto, se me había pasado comentar que Gonzalo, Antonio y José se sienten muy agradecidos con las personas que les dieron posada en el hotel, ya que si de no haber sido así, tal vez no hubiéramos llegado a alguna de las conferencias ya que estaba muy retirado el lugar donde nos hospedamos.

Después de la conferencia de PostgreSQL y de la de Guido de Python, empezaron a hacer entrega de las constancias del congreso y de los talleres. Como en ese momento Rasmus ya se tenía que ir, aprovechamos y nos tomamos unas fotos con él.

Por la tarde, llego una de las conferencias mas esperadas, la conferencia de miguel, en ella nos mostró como Mono ha ido avanzando en el campo de los videojuegos, en el campo de la programación haciendo Mono cada vez mas compatible con los programas creados en .NET, nos habló brevemente sobre los problemas que tuvo Compiz y porque de la creación de Beryl y después le dio paso Xavi que nos presentó LinuxShow. LinuxShow es una muestra divertida de conceptos informáticos con un buen toque

de humor circense.

Durante el transcurso del congreso se llevo a cabo el campeonato de tae kwon do en la sala de junto a donde se estaban llevando las conferencias, debido a eso, estuvimos escuchando los gritos que a cada rato nos causaba uno que otro sobresalto pero pues Miguel y Xavi se encargaron de darle el toque humorístico a la situación y haciendo participe a los asistentes como por ejemplo, poner a gritar a alguien como si se fuera a morir ¡pero con el micrófono! fue realmente divertido.

Después de que nos tomamos fotos con todo el que se pudo, dimos por terminado el congreso y nos preparamos para regresar nuevamente a Chetumal.

Como siempre, realmente fascinantes al reuniones que se hacen a través de la comunidad.

RevistaSL agradece todo el apoyo recibido por parte de Miguel para cubrir el evento. Mas información:

http://www.gulev.org.mx

Matt Dillon

El "urkonn" entrevista al lider de DragonFly

Julio Acuña Carrillo julio.acuna@revista-sl.org

RSL: ¿Qué es DragonFlyBSD y cuál es su estado actual?

MD: DragonFly es parte de la familia de sistemas BSD UNIX. Actualmente tenemos una distribución completamente integrada de kernel y sistema y la mayoría de las principales aplicaciones de usuario compilan e instalan desde PkgSrc. Corre en plataformas PC (en modo de 32 bits).

RSL: ¿Cómo comenzó DragonFlyBSD, fue solamente un desacuerdo con la dirección que estaba tomando FreeBSD o tenías otros intereses que te condujeron a la creación de una nueva distribución BSD?

MD: Ambas cosas, de hecho. Tenía un gran desacuerdo con la manera en la que FreeBSD estaba implementando el soporte para SMP (multi procesador simétrico).

Sentí que las metodologías y el código no estaban siendo hechos en una forma sustentable y que serían propensos a bugs difíciles de encontrar y debilitamiento del código.

Pero también tenía otros intereses, y la combinación de las dos cosas resultaron en el inicio por mi parte del proyecto DragonFly.

El mayor objetivo de DragonFly es integrar completamente capacidades de clustering dentro del kernel, para poder clusterizar confiablemente las máquinas juntas y tener acceso a todos sus recursos reunidos sin esfuerzo. El proyecto ha existido por dos años y ahora estamos aproximadamente a dos años de completar nuestro objetivo.

RSL: Creo que cada sistema operativo tiene su propia filosofía, ¿cuál es la filosofía detrás de DragonFlyBSD?

MD: Guau. Bueno, puedo hablar por horas acerca de mi filosofía personal, pero en breve, aparte de querer completar los objetivos que he fijado para el proyecto, quiero crear un ambiente de desarrollo social que quite las barreras entre los desarrolladores en lugar de construirlas. Eso significa tanto reconocer las habilidades individuales como reconocer que el núcleo del proyecto... su código base, debe ser considerado un recurso de la comunidad y nunca ser dividido o "poseído" por una persona en particular. Algunas veces los proyectos llegan a ser excesivamente estratificados y perderse en requerimientos que sólo sirven a los intereses y deseos de una pequeña parte de la comunidad – a veces creo que el proyecto FreeBSD ha sufrido bastante de ésto en los últimos años-- no quiero en absoluto que DragonFly se convierta en eso.

Una de las grandes ventajas de la comunidad BSD es que muchos...

del código base del SO, si no compatible, es todavía sencillo de portar entre los proyectos. Me aseguro de animar a todos los desarrolladores de DragonFly a que vean el trabajo hecho en todos los proyectos BSD, así como en Linux, y traer ese trabajo a DragonFly cuando es claramente superior. Hemos traído infraestructura considerable de OpenBSD, NetBSD, FreeBSD, incluso un poco de Linux, y por supuesto mucho de GNU y hemos reescrito igual cantidad de infraestructura nativamente.

Otra de las filosofías principales del proyecto es no traer código antes de tiempo. Todos queremos SMP, pero no estamos deseosos de hacer hacks rápidos al código base que sólo nos comerá en el futuro o crear tal desastre que esencialmente se vuelve inmantenible y propenso a bugs. Esto es particularmente cierto para nuestro soporte SMP. Un buen soporte SMP es una de las metas del proyecto y tenemos una metodología de desarrollo para eso, pero no vamos a apresurarnos a través de hacks que desestabilicen el sistema sólo para ganar unos cuantos benchmarks. Mi filosofía, la cual todos en el proyecto comparten, es usar el algoritmo correcto "PRIMERO", incluso cuando eso signifique la reescritura del subsistema entero, y después trabajar hacia el objetivo desde la base en lugar de en base a un algoritmo de hace décadas que nunca tuvo en mente al SMP (o al clustering).

Desde hace dos años casi cada subsistema en el kernel de DragonFly se ha reescrito con esto en mente, y cuando a algunos les parece que que no estamos progresando hacia nuestras metas, de hecho estamos haciendo un progreso significativo. Estoy muy contento con la manera en la que el proyecto está progresando.

RSL: ¿Por qué alguien debería escoger DragonFlyBSD, cuáles son sus principales ventajas?

MD: Para producción en masa tal vez sea mejor esperar a que nos acerquemos a nuestro principal objetivo. En ese punto DragonFly tendrá distintas ventajas para una gran clase de problemas sobre otros sistemas operativos.

Por supuesto, DragonFly trabaja bien ahora como plataforma tipo-UNIX genérica, similar a Linux (y de hecho, BSD original ha existido mas tiempo que Linux), pero como Linux ha tomado ese mercado, yo estaría reticente a recomendar DragonFly sobre Linux para cualquier tipo de aplicaciones comerciales importantes. Todavía.

Sin embargo, a cualquiera con requerimientos mas personales, un pequeño negocio o la necesidad de una pequeña plataforma PC, y para saber cómo correr un sistema UNIX, probablemente le gustará DragonFly como está justo ahora. Lo que quiero realmente es un deployment modesto el cual podamos soportar con nuestros limitados recursos de desarrolladores. Ese fue quizá un enunciado demasiado directo, pero siento que siempre es importante entender y vivir en el mundo real. No siento que sea posible que Linux se vaya con todo el mercado simplemente porque mucho del mercado esta basado mas en la infraestructura de aplicaciones de usuario (basado en estándares abiertos los cuales todas las distros de Linux y BSD soportan) que en la base del sistema operativo. La intención con DragonFly es permanecer dentro del lado de las aplicaciones mientras se añaden características importantes al sistema base.

RSL: No mucha gente conoce la Amiga en estos días (es una pena), entiendo que estuviste involucrado con ella hace algunos años, DragonFlyBSD está tomando ideas del AmigaOS o cualquier otro sistema operativo?

MD: No mucho ya, pero ciertos conceptos usados en DragonFly, particularmente la capa de mensajería del kernel, fueron desarrollados a partir de mis experiencias con la Amiga. Creo que lo que mas me dejaron mis años con la Amiga es pragmatismo con respecto a la longevidad del código. Ideas, algoritmos y protocolos son para siempre, el código (en forma usable) no. Los principales objetivos de DragonFly son orientados a la idea, algoritmo y protocolo, y si llego apropiadamente a esas ideas, algoritmos y protocolos, todavía se conocerán en cien años incluso si el código original usado para implementarlos se vuelve irreconocible...

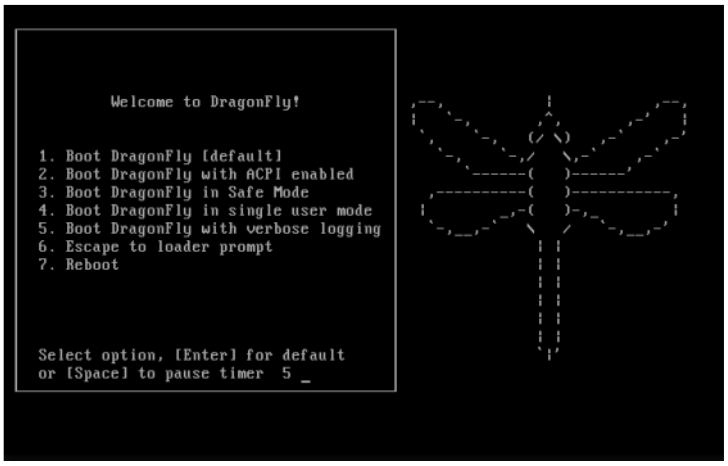
para futuras generaciones.

RSL: ¿Cómo describirías el kernel de DragonFlyBSD?

MD: Progresando bien hacia nuestros objetivos.

RSL: Al tiempo que esta entrevista se publique tal vez la próxima liberación de DragonFlyBSD esté lista para descargar, ¿nos podrías decir cuáles son los cambios principales de la versión 1.6?

MD: La liberación 1.8 está planeada para fin de enero. La entrada del Diario para enero de 2007 en http://www.dragonflybsd.org/status/diary.shtml contiene una lista completa de los cambios.



RSL: Ahora DragonFlyBSD está usando el pkgsrc de NetBSD para instalar software de terceros, ¿hay planes para crear un sistema de paquetes específicamente para DragonFlyBSD?

MD: Planeamos seguir usando PkgSrc. Hay muchas razones para esto pero sólo dos para destacan. Primero, los desarrolladores de PkgSrc están dedicados a producir una infraestructura portable, lo que significa que no tenemos que hackear constantemente paquetes o DragonFly para mantener la compatibilidad (algo que a veces teníamos que hacer con el sistema de ports de FreeBSD conforme DragonFly divergía de FreeBSD). Los paquetes actualizados por desarrolladores de NetBSD u OpenBSD funcionan con DragonFly y en la ocasión en la que algún problema surge, se trabaja en una solución basada en el objetivo de la portabilidad. En segundo lugar, PkgSrc es un medio muy eficiente con respecto al tiempo y esfuerzo para proveer a DragonFly con una gran cantidad de paquetes de terceros.

Como proyecto pequeño, no podemos dedicar un gran número de recursos a mantener paquetes y ser aún capaces de trabajar hacia los objetivos del sistema operativo.

PkgSrc no es perfecto, actualizar paquetes todavía es una molestia, pero se ajusta perfectamente con los objetivos y filosofía del proyecto DragonFly como un guante.

RevistaSL agradece a Matt Dillon por tu tiempo y facilidades para esta entrevista; de las filas de RevistaSL enviamos nuestros mejores deseos para que el proyecto DragonFly siga avanzando por tan buen rumbo.

Mas información:

http://www.dragonflybsd.org

EVENTOS SL
la agenda de los geeks

Victor Hugo Cordova buzon@revista-sl.org

FLOSS International Conference 2007
Marzo 7-9

El FIC tiene como objetivo ser un marco de encuentro para las principales iniciativas relacionadas con los FLOSS, incidiendo especialmente en aquellas relacionadas con la Unversidad de Cádiz. El FLOSS cubrirá las lineas de educación, tecnología e investigación.

Este primero congreso internacional se celebrerá en la Facultad de Ciencias Sociales y de la Comunicación, ubicada en el campus de Jerez de la Frontera perteneciente a la Universidad de Cádiz.

Mas información en el sitio web: http://softwarelibre.uca.es/jornadas/fic

Congreso Nacional de Software Libre 2007
Febrero 13-16

El CONSOL este año se celebrará en las instalaciones de la Facultad de Ingeniería de la Universidad Nacional Autónoma de México, en Ciudad Universitaria.

Las actividades comprenden conferencias, talleres y conferencias magistrales. Todo ello con la presencia de las maximas personalidades del software libre en México y el mundo.

Mas información: http://www.consol.org.mx

BarCamp Ciudad de México
Febrero 17-18

Software libre está fundado en compartir. No sólo compartir código, sino también ideas, conceptos y soluciones. Sitios como Sourceforge y Freshmeat son importantes para distribuir el código de software libre, pero no debemos olvidar la importancia de distribuir conocimiento a través de un diálogo entre las personas involucradas en su creación.

Los congresos tradicionales son muy buenos para introducir a mucha gente en nuevos temas, pero a veces es necesario juntarnos en grupos pequeños para platicar y discutir en un ambiente informal. BarCamp México es una no-conferencia creada para compartir cualquier conocimiento relacionado con el software libre.

Mas información: http://barcamp.org/BarCampMexico

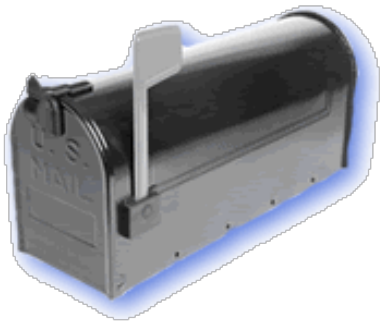
¿Conoces algún evento relacinado con el código libre y las nuevas tecnologías?... notificanoslo. Si eres un organizador convierte a RevistaSL en el medio electronico para dar a conocerlo.

Escribe a buzon@revista-sl.org

BUZÓN SL

nuestros lectores opinán

Victor Hugo Cordova **buzon@revista-sl.org**



Un tip debianero...!

Bueno el presente tip es para recuperar una instalación de debian, debido a que se borro el MBR o (sector de arranque maestro) por culpa de "Wintendo XP", para recuperar el sistema usaremos Knoppix, yo tengo la 3.3 los pasos son muy sencillos no fue necesario chrootear la particion raiz montar nada, es mucho mas simple.

1.- booteamos del cd de knoppix, una vez dentro del sistema abrimos una terminal y en ella realizamos el siguiente procedimiento.

Nota: este procedimiento es solo para grub, ya que no he trabajado con los otros gestores de arranque

```
# grub

grub> root (hd0,4) (en mi caso es asi, si no sabes cual es la tuya usa  vi que trae
knoppix y anda a /mnt/hdax/boot/grub/menu.lst, ahi sale en root)

grub> setup (hd0)

grub> quit
```

salu2

Mikel Carozzi
http://www.psynova.cl
cadrogui@gmail.com

RevistaSL quiere saber quienes nos leen... envia tus comentarios, opiniones, sugerencias, tips, etc.

Escribe a buzon@revista-sl.org

No te quedes en el anonimato, ¡ESCRIBENOS!

RevistaSL

El software libre hecho revista

Colaboradores SL

Longino Jacome
Cesár Yañez Fernández
Gonzalo Martínez
Martin Marquez
Andres Vargas
Lael Gonzalez Rosales
Matt Dillon
Mikel Carozzi

Comunidad Linux Chetumal
www.linux-chetumal.org.mx

RTM Security Team
www.zonartm.org

Grupo de Usuarios Linux del Estado de Hidalgo
www.guleh.org

Grupo de Usuarios de Software Alternativo de Toluca
www.gusat.org

Grupo de Usuarios Open Source de Cancún
www.tucancunix.net

Colectivo Estudiantil de Química UNAM
colectivoquimica.dnsalias.org

Laboratorio de Investigación y Desarrollo de Software Libre
www.lidsol.org

Bytez SL #~

Un número mas de RevistaSL concluye, en lo particular esta edición me ha gustado bastante, el equipo editorial en esta ocasión trabajo mas unido que nunca, pero como siempre nuestros lectores son quienes tienen la última palabra.

Desde este momento quiero hacer la invitación a que participen en RevistaSL Num. 7 "Negocios y Software Libre"; recuerden que pueden enviar todo lo que ustedes deseen, RevistaSL es un proyecto de divulgación con el fin de mostrar lo que la comunidad esta haciendo.

Ademas les pedimos su apoyo para que este proyecto llegue a mas personas, que no solo sea una publicación de amantes del Software Libre para amantes del Software Libre, queremos que RevistaSL cumplá su misión de divulgación y para ello pedimos su apoyo en adquisición de publicidad o promoción del proyecto.

Le doy mi agradecimiento a los colaboradores, a nuestros patrocinadores y a la Oficina de Software Libre de la Universidad de Cádiz, a quienes apoyaremos con todo lo que este en nuestras manos.

Muchas gracias a todos y nos leeremos el próximo número.

Carlos A. Lozano Vargas
Coordinación Editorial SL