Manual de FreeBSD

Proyecto de Documentacion de FreeBSD

doc@FreeBSD.org

Manual de FreeBSD
por Proyecto de Documentacion de FreeBSD

Publicado Febrero 1999

Bienvenido a FreeBSD!. Este manual cubre la instalación y uso diario de *FreeBSD Release 4.0*. Este manual está en *constante evolución* y es el resultado del trabajo de muchas personas. Algunas secciones no están completas y otras no están actualizadas. Si estás interesado en colaborar en este proyecto, envía un mail a Lista del Proyecto de Documentación de FreeBSD <`freebsd-doc@FreeBSD.ORG`>. La última versión de este documento está siempre disponible en el servidor World Wide Web de FreeBSD (http://www.FreeBSD.ORG/). También está disponible en texto (handbook.latin1), postscript (handbook.ps) o HTML (handbook-html.tar.gz) vía HTTP o comprimido con gzip desde el servidor FTP de FreeBSD (ftp://ftp.FreeBSD.ORG/pub/FreeBSD/docs/) o uno de sus numerosos mirrors. También puedes realizar búsquedas en el manual (http://www.FreeBSD.ORG/search.html).

Tabla de contenidos

I. Empezamos

Capítulo 1. Introducción

FreeBSD es un sistema operativo basado en 4.4BSD-Lite para arquitectura Intel (x86). Para una descripción de FreeBSD, mira FreeBSD en una palabra. Para conocer la historia del proyecto, lee una breve historia de FreeBSD. Para ver la descripción de la última release, lee sobre la release actual. Si estás interesado en contribuir de alguna manera al proyecto FreeBSD, (código, equipamiento, donaciones, etc), por favor, lee la sección contribuyendo a FreeBSD.

1.1. FreeBSD en una palabra

FreeBSD es un sistema operativo para ordenadores personales basado en CPU's de arquitectura Intel, incluyendo procesadores 386, 486, y Pentium (versiones SX y DX). También son soportados los procesadores compatibles Intel como AMD y Cyrix. FreeBSD te ofrece muchas características avanzadas antes solo disponibles en ordenadores mucho más caros. Estas características incluyen:

- *Preemptive multitasking* con ajuste dinámico de prioridades para asegurar la mejor compartición de recursos entre aplicaciones y usuarios.
- *Acceso Multiusuario* significa que diferentes personas pueden usar un sistema FreeBSD simultaneamente para realizar diferentes trabajos. Los periféricos del sistema como impresoras y cintas también pueden ser compartidas entre todos los usuarios del sistema.
- Completa *conectividad TCP/IP* incluyendo soporte SLIP, PPP, NFS y NIS. Esto significa que tu máquina FreeBSD puede comunicarse facilmente con otros sistemas, además de actuar como servidor principal, proveyendo de funciones vitales como NFS (acceso a ficheros remotos), servicios de correo electrónico o poner a tu organización en Internet con WWW, ftp, router, firewall (seguridad).
- *La protección de memoria* asegura que las aplicaciones (o usuarios) no puedan interferirse unos con los otros. En caso de que una aplicación falle, no afectará al resto de aplicaciones en funcionamiento.
- FreeBSD es un *sistema operativo de 32 bits* y fue diseñado así desde el primer momento.
- El sistema estandar en la industria *X Window* (X11R6) provee una interficie gráfica de usuario (GUI) para las tarjetas VGA y monitores más comunes incluyendo todo el código fuente.
- *Compatibilidad binaria* con muchos programas nativos de SCO, BSDI, NetBSD, Linux y 386BSD.
- Cientos de aplicaciones *ready-to-run* están disponibles en las colecciónes de *ports* y *packages*. Porqué buscar en la red cuando puedes encontrarlo todo aquí?.
- Miles de aplicaciones *fáciles de portar* disponibles en Internet. FreeBSD es compatible con el código fuente de los más populares y comerciales sistemas Unix, y las aplicaciones requieren unos mínimos cambios (si es que lo requieren) para compilar.

- *Memoria virtual* paginada bajo demanda satisface eficientemente a las aplicaciones con mucho consumo de memoria, manteniendo aun respuestas interactivas al resto de usuarios.
- *Librerís compartidas* (el equivalente en Unix a las DLLs de Windows) que ofrecen un uso eficiente del espacio en disco y memoria .
- Se incluye un completo conjunto de herramientas de desarrollo en *C*, *C++* y *Fortran*. Muchos lenguajes adicionales para investigación avanzada y desarrollo están incluidos en las colecciones de ports y packages.
- *Código fuente* completo de todo el sistema, ofreciendote el máximo grado de control sobre tu entorno. Porqué estar bloqueado en una solución propietaria cuando puedes tener un verdadero sistema abierto?.
- Extensa *documentación on-line* .
- *Y mucho más!*

FreeBSD está basado en la release 4.4BSD-Lite del Computer Systems Research Group (CSRG) de la Universidad de California en Berkeley, siguiendo la tradición que ha distinguido el desarrollo de los sistemas BSD. Además del buen trabajo realizado por el CSRG, el proyecto FreeBSD ha invertido miles de horas en ajustar el sistema para ofrecer las máximas prestaciones en situaciones de carga real.

Las aplicaciones a las que se puede someter FreeBSD están solo limitadas por tu imaginación. Desde desarrollo de software hasta automatización o robótica, control de inventarios o correcciones de azimuts de antenas de satélite remotas; si puede hacerse con un producto Unix comercial, es más que seguro que puede hacerse con FreeBSD. FreeBSD se beneficia significantemente de las miles de aplicaciones de alta calidad desarrolladas por centros de investigación y universidades de todo el mundo, disponibles a un coste mínimo o sin coste alguno. Las aplicaciones comerciales también están disponibles apareciendo en mayor número cada día.

Gracias a que disponemos de todo el código del sistema, éste puede ser personalizado para aplicaciones o proyectos especiales de maneras que generalmente no son posibles con la mayoría de los sistemas operativos comerciales. Aquí tienes algunos ejemplos de aplicaciones de FreeBSD:

- *Servicios Internet:* El robusto stack TCP/IP integrado en FreeBSD lo hace una plataforma ideal para una gran cantidad de servicios Internet como:
  - Servidores FTP
  - Servidores WWW
  - Servidores Gopher
  - Servidores de Correo electrónico
  - News USENET
  - Sistema de BBS
  - Y mucho más...

  Puedes empezar facilmente con un ordenador PC 386 e ir actualizandolo a medida que crezcan tus necesidades.

- *Edicación:* Eres un estudiante de informática o de algun campo relacionado con la ingeniería?. No hay mejor manera de aprender sobre sistemas operativos, arquitectura de computadores y redes que poder "poner las manos" en el código completo de un sistema operativo como FreeBSD. Cantidad de packages libremente disponibles sobre CAD, matemáticas y diseño gráfico hacen de este sistema la herramienta ideal para aquellos que usan los ordenadores para *otras* cosas.
- *Investigación:* Con todo el código fuente disponible, FreeBSD es una excelente plataforma de investigación en sistemas operativos. La naturaleza de libre distribución de FreeBSD ha hecho posible que grupos remotos hayan podido trabajar y colaborar en desarrollos compartidos sin tener que preocuparse de licencias especiales o limitaciones de ningún tipo.
- *Networking:* Necesitas un nuevo router?. Un servidor de nombres (DNS)? Un firewall para mantener a la gente fuera de tu red interna?. FreeBSD puede convertir esos antiguos PC's 386 o 486 en routers avanzados con capacidades avanzadas de filtrado de paquetes.
- *Estación de trabajo X Window:* FreeBSD es una buena elección para una solución de terminales X baratos, usando el servidor libre XFree86 o uno de los excelentes servidores comerciales producidos por X Inside. FreeBSD te permite ejecutar localmente las aplicaciones además de ejecutar las del servidor central.
- *Desarrollo de software:* El sistema básico de FreeBSD incluye un complemento completo de herramientas de desarrollo incluyendo el compilador y debugger GNU C/C++.

FreeBSD está disponible en formato binario o código fuente en CDROM y vía ftp anónimo. Mira Obteniendo FreeBSD para más detalles.

1.2. Una breve historia de FreeBSD

Contribuido por Jordan K. Hubbard <`jkh@FreeBSD.org`>.

El proyecto FreeBSD tuvo nació en los inicios de 1993 por una escisión parcial de los 3 coordinadores del "Unofficial 386BSD Patchkit": Nate William, Rod Grimes y yo mismo.

Nuestro objetivo original era producir una muestra intermedia de 386BSD para solucionar una serie de problemas que el mecanismo del patchkit no era capaz de solucionar. Alguno de vosotros quizás recuerde el nombre inicial del proyecto "386BSD 0.5" o "386BSD Interim".

386BSD era el sistema operativo de Bill Jolitz y había estado muy desatendido durante todo un ano. Al sentirse todo el patchkit cada vez más desanimado, decidimos que había que hacer algo e intentamos ayudar a Bill haciendo esta muestra sin fallos. Estos planes se vieron frenados cuando Bill decidió de repente dejar de formar parte del proyecto.

No nos costó demasiado decidir seguir adelante, aún sin el soporte de Bill, así que adoptamos el nombre de "FreeBSD", sugerido por David Greenman. Nuestro objetivo inicial era consultar con los usuarios actuales del sistema, y, una vez aclaradas las cosas, intentar que el proyecto se convirtiese en una realidad. Contacté con Walnut

Creek CDROM con la intención de disponer de un canal de distribución para todos aquellos que no tuvisen un fáil acceso a Internet. Walnut Creek no sólo apoyó la idea de distribuir FreeBSD en CDROM si no que aportó una máquina sobre la que desarrollar los proyectos y una rápida conexión a Internet. Sin el apoyo y la confianza que Walnut Creek depositó en un proyecto desconocido y recién nacido, es seguro que FreeBSD no hubiese podido llegar tán lejos y tán rápido como lo ha hecho.

Durante esta época, aparecieron una serie de nubarrones inesperados en el horizonte ya que Novell y la U.C. Berkeley solucionaron su larga pugna legal sobre el estatus del Net/2 de Berkeley. Una de las condiciones del acuerdo fue la concesión por parte de la U.C. Berkeley de que Novell se hiciese cargo de gran parte del código de Net/2, ya que de hecho la había adquirido anteriormente a AT&T. Lo que Berkeley recibió a cambio fue el permiso de declarar libre la release de 4.4BSD-Lite, y que todos los usuarios existentes de Net/2 serín conminados a migrar de sistema. Esto incluyó a FreeBSD, y el proyecto obtuvo de plazo hasta Julio de 1994 para terminar de ofrecer su producto basado en Net/2. Bajo los términos de este acuerdo, se le permitió al proyecto una última release, FreeBSD 1.1.5.1.

A partir de ese momento, FreeBSD se dedicó a la árdua tarea de, literalmente, reinventarse a si mismo, desde un nuevo e incompleto 4.4BSD-Lite. El proyectó terminó esta transición en Diciembre de 1994, y, en Enero de 1995 se publicó la release FreeBSD 2.0 en Internet y en CDROM. Teniendo en cuenta todos los problemas, la release obtuvo un éxito importante, seguida de la más robusta y fácil de instalar FreeBSD 2.0.5 en Junio de 1995.

En Agosto de 1996 se publicó la release 2.1.5, consiguiendo ser suficientemente conocida entre ISP y comunidades comerciales. La release 2.1.7.1 en Febrero de 1997 se convirtió en el final del desarrollo de la rama 2.1-stable. En estos momentos, ésta rama se encuentra en modo de mantenimiento, realizando sólo trabajos de seguridad o solución de problemas críticos

FreeBSD 2.2 nació de la línea principal de desarrollo (“-current”) en Noviembre de 1996 como la rama RELENG_2_2, y la primera release completa se realizó en Abril de 1997. Las siguientes releases de la rama 2.2 fueron en verano y otono de 1997, apareciendo la última en Julio de 1998. La primera release oficial de la rama 3.0 apareció en Octubre de 1998, siendo publicada la última release de la rama 2.2 (2.2.8) en Noviembre de 1998.

Desarrollos a largo plazo como el soporte SMP o de la plataforma DEC de ALPHA continuará en la rama 3.0-current (ya 4.0-current) y SNAPshots de la 3.0 en CDROM /y, por supuesto, en la red).

1.3. Objetivos del Proyecto FreeBSD

Contribuido por Jordan K. Hubbard <jkh@FreeBSD.org>.

Los objetivos del Proyecto FreeBSD son proveer software que pueda ser usado para cualquier objetivo sin que ésto acarree una carga. Muchos de nosotros tenemos bastante que ver con el código (y proyecto). Creemos que nuestra primera y más importante “misión” es proveer código a cualquier persona de manera que el código pueda ser muy distribuido y ofrezca el mayor número posible de beneficios. Este es, creo, uno de los objetivos principales del Software Libre, al que apoyamos entusiastamente.

El código incluido en nuestra distribución que se encuentra bajo licencia GPL, GNU o GLPL tiene algunas restricciones más. Debido a los problemas que puede acarrear el uso de software de licencia GPL de manera comercial, invitamos a reemplazar este software por otros bajo licencias mas relajadas como BSD, siempre y cuando sea posible.

1.4. El modelo de desarrollo de FreeBSD

Contribuido por Satoshi Asami <asami@FreeBSD.org>.

El desarrollo de FreeBSD es un proceso muy abierto y flexible, siendo, literalmente, creado a partir de las contribuciones de cientos de personas de todo el mundo, como se puede ver en nuestra lista de participantes. Estamos constantemente en la búsqueda de nuevos desarrolladores e ideas, y aquellos interesados en estar más envueltos en el proyecto, sólo necesitan ponerse en contacto con nosotros a través de la lista Lista de discusiones técnicas de FreeBSD <freebsd-hackers@FreeBSD.ORG>. Aquellos que prefieran trabajar más independientemente también son bienvenidos, pudiendo usar las facilidades de nuestro servidor FTP en ftp.freebsd.org (ftp://ftp.freebsd.org/pub/FreeBSD/incoming) para distribuir sus propios parches o código fuente de trabajos en progreso. La lista Lista de anuncios de FreeBSD <freebsd-announce@FreeBSD.ORG> está disponible para comunicar a otros usuarios de FreeBSD las áreas más importantes de trabajo.

Cosas útiles de conocer sobre el Proyecto FreeBSD y sus procesos de desarrollo, ya sea de manera independiente o cooperación en grupos:

El depósito CVS

> El árbol de código fuente principal es mantenido por CVS (http://www.cyclic.com/cyclic-pages/CVS-sheet.html) (Concurrent Version System), una herramienta de libre disposición para el control de código fuente que está incluida en FreeBSD. El CVS principal (http://www.freebsd.org/cgi/cvsweb.cgi) reside en un servidor de Concord, California, desde el que se replica a numerosos servidores mirror repartidos por todo el mundo. El árbol CVS, así como las ramas -current y -stable pueden ser facilmente replicadas en tu propia máquina. Por favor, mira en la sección Sincronizando tu árbol de código fuente para más información sobre este sistema.

La lista de "committers"

> Los committers son las personas que tienen acceso de *escritura* al CVS, y están autorizados para realizar modificaciones en el código fuente (el término “committer” proviene del comando cvs(1) `commit`, usado para incorporar los nuevos cambios en el CVS). La mejor manera de hacer envíos para que sean revisados por los committers es el comando send-pr(1). También puedes enviar un mail a la dirección <committers@freebsd.org>.

El core-team de FreeBSD

El FreeBSD core team será el equivalente al conjunto de directores si FreeBSD fuese una compañía. La tarea principal del core team es asegurar que el proyecto, como una unidad, está en el buén camino y apuntar hacia la direcció correcta. Invitar a los desarrolladores dedicados y responsables a participar en el grupo de committers es otra de las funciones del core team, así como incorporar nuevos miembros al mismo. Muchos de los miembros del core team empezaron como committers, y a los que la adicción al proyecto ha conseguido lo mejor de ellos.

Algunos miembros del core team también tienen algunas áreas de responsabilidad específicas, significando que se encargan de asegurar que partes importantes del sistema funcionen como se espera.

> **Nota:** La mayoría de miembros del core team son voluntarios y no se benefician del proyecto economicamente. La analogía de “equipo de directores” actualmente no es demasiado acertada, y es más justo decir que estas personas son las personas que dan sus vidas en favor de FreeBSD y en contra de su sano juicio *;)*

Contribuyentes externos

El más importante grupo de desarrolladores son los propios usuarios, ya que nos ofrecen sus idea, aportaciones, parches, código, etc. La principal manera de participar en el desarrollo de FreeBSD es subscribirse en la lista Lista de discusiones técnicas de FreeBSD <`freebsd-hackers@FreeBSD.ORG`> (mira en la información de listas) donde se discuten este tipo de cosas.

La lista de aquellos que han contribuido con alguna parte de código es muy larga y creciendo cada día, así que porqué no participar en ella y contribuir con algo hoy mismo a FreeBSD? *:-)*

Enviar código no es la única manera de contribuir al proyecto; para una lista más completa de cosas por hacer, por favor, mira en la sección cóomo contribuir de este manual.

En definitiva, nuestro modelo dedesarrollo está organizado como una serie de círculos concéntricos. El modelo centralizdo está disenado para la conveniencia de los *usuarios* de FreeBSD, que tienen la posibilidad de seguir un código fuente centralizado, sin dejar fuera a los potenciales participantes y contribuyentes de código. Nuestro deseo es presentar un sistema operativo estable con una larga y coherente serie de aplicaciones que sean sencillos de instalar y usar, y este modelo funciona muy bien para conseguir estos objetivos.

Todo lo que le pedimos a aquellos que quieran participar como desarrolladores de FreeBSD es la misma dedicación que tienen los desarrolladores actuales para poder seguir trabajando de la misma manera que se ha hecho hasta ahora.

1.5. Sobre la Release Actual

FreeBSD es un sistema de libre disponibilidad, con el código fuente completo y basado en 4.4BSD-Lite para máquinas Intel i386/i486/Pentium/PentiumPro/Pentium II (o compatibles). Está basado en el software del grupo CSRG de la U.C. Berkeley, con algunas mejoras de NetBSD, OpenBSD, 286BSD y la Free Software Foundation.

Desde nuestra release FreeBSD 2.0 en Enero de 1995, el rendimiento, características y estabilidad de FreeBSD ha aumentado considerablemente. El cambio más importante es el sistema de memoria virtual con un buffer de cache VM/file que no sólo incrementa el rendimiento, si no que además reduce el consumo de memoria, haciendo que el sistema sólo necesite 5Mb de memoria en su configuración mínima. Otras mejoras incluyen soporte completo de cliente y servidor NIS, soporte de transacciones TCP, dial-on-demand PPP, un nuevo subsistema SCSI, soporte de RDSI, soporte de tarjetas FDDI y Fast Ethernet, soporte de la tarjeta ADAPTEC 2940, y cientos de soluciones en errores anteriores.

También hemos tomado muy en cuenta los comentarios y sugerencias de nuestros usuarios para hacer lo que creemos es un sistema de instalación sencillo y eficiente. Tus comentarios sobre este proceso son bienvenidos, ya que es un trabajo en constante evolución.

Además de las distribuciones base, FreeBSD ofrece una nueva colección de software portado con cientos de programas. A finales de Agosto de 1998 estabamos en más de 1700 ports!. La lista de ports va desde servidores http, hasta juegos, lenguajes de programación, editores y cualquier otra cosa que te puedas imaginar. La colección de ports completa requiere alrededor de 26Mb de espacio, siendo todos los ports “deltas” a sus códigos fuente originales. Esto hace que para nosotros sea mucho más sencillo actualizar los ports, y reduce considerablemente el espacio en disco requerido. Para compilar un port, simplemente necesitas cambiar al directorio del programa que quieres instalar, y teclear `make all` seguido de `make install` después de la compilación dejando al sistema que haga el resto. La distribución original completa de cada port se obtiene dinámicamente del CDROM o un servidor local FTP, de manera que sólo necesitas el espacio de disco necesario para los ports que quieras instalar. La mayoría de ports también se pueden obtener en formato precompilado, llamado “package”, que puede ser instalado con un simple comando (pkg_add) para aquellos que no quieran compilar sus propios ports a partir del código fuente.

Documentación adicional sobre el proceso de instalación y uso de FreeBSD la puedes encontrar en el directorio `/usr/share/doc` en cualquier máquina con FreeBSD 2.1 o posterior. También puedes ver los manuales instalados de manera local con cualquier navegador HTML usando las siguientes URLs:

El Handbook de FreeBSD

file:/usr/share/doc/handbook/handbook.html

Las FAQ de FreeBSD

file:/usr/share/doc/FAQ/FAQ.html

También puedes visital las copias centrales (y normalmente más actualizadas) en http://www.freebsd.org.

La distribución principal de FreeBSD no contiene el código DES, al estar prohibida su exportación fuera de USA. Hay un package, para usar sólo en USA, que contiene los programas que normalmente usa DES. Una distribución DES europea y libremente distribuible fuera de USA, también está disponible, cóomo se describe en las FAQ (../FAQ/FAQ.html).

Si todo lo que necesitas es seguridad de passwords en FreeBSD, y no tienes necesidad de copiar passwords encriptados desde diferentes máquinas (Sun, DEC, etc) al fichero de passwords de FreeBSD, entonces la seguridad basada en MD5 que te ofrece FreeBSD es lo que necesitas. Creemos que nuestro modelo de seguridad no tiene nada que perder con respecto a DES, y sin ningún problema de restricción de exportación. Si estás fuera (o incluso dentro) de USA, pruébalo!.

Capítulo 2. Instalando FreeBSD

Así que te gustaría probar FreeBSD en tu máquina?. Esta sección es una guía rápida sobre lo que necesitas hacer. FreeBSD puede ser instalado desde una gran cantidad de soportes incluyendo CDROM, floppies, cintas magnéticas, una partición MS-DOS y, si tienes conexion de red, vía FTP anónimo o NFS.

Sea cual sea el soporte de instalación que elijas, puedes empezar por crear *los discos de instalación* como se describe más abajo. Arranca tu ordenador con el instalador de FreeBSD, aunque no estés pensando en realizar la instalación, y podrás obtener gran cantidad de información sobre la compatibilidad del hardware de tu ordenador y FreeBSD, además de una descripción más exacta sobre las diferentes posibilidades de instalación.

Si tienes pensado hacer la instalación vía FTP, sólo necesitas el disco de arranque, ya que él solo se encarga de gestionar todo lo referente a la conexión, ya sea ethernet o PPP.

Para más información sobre cómo obtener la última distribución de FreeBSD, por favor, mira la sección Obteniendo FreeBSD en el Apéndice.

Bién, para empezar a caminar, sigue los siguientes pasos:

1. Revisa la sección configuraciones soportadas de esta guía de instalación para asegurarte de que tu hardware es soportado por FreeBSD. Sería de gran ayuda que hicieses una lista de cualquier tarjeta especial que tengas instalada, como controladoras SCSI, tarjetas de red o tarjetas de sonido. Esta lista debería incluir parámatros relevantes de configuración como interrupciones (IRQ) y direcciones de entrada/salida (IO ports).
2. Si estás instalando FreeBSD desde un CDROM tienes diferentes opciones de instalación:
   - Si el CD ha sido grabado con el soporte de arranque El Torito, y tu sistema soporta arranque directo desde CDROM (muchos sistemas antiguos *no*), simplemente inserta el CD en el lector y arranque directamente desde él.
   - Si estás ejecutando DOS y tienes los drivers adecuados para acceder al lector de CDROM, ejecuta el fichero install.bat incluido en el CD. Este intentará arrancar la instalación de FreeBSD desde DOS.

     Nota: Debes hacer esto desde el DOS y no desde una máquina Windows.

     Si quieres instalar FreeBSD desde una partición DOS (por ejemplo, por que FreeBSD no soporta tu lector de CDROM), ejecuta primero el programa setup para copiar los ficheros adecuados del CD a la partición. A continuación ejecuta el programa de instalación.

   - Si cualquiera de los dos métodos anteriores funciona, puedes pasar por alto el resto de esta sección, en caso contrario, tu opción final es crear un disco de arranque con la imágen `floppies\boot.flp`. Salta al paso 4 para instrucciones sobre como hacerlo.

3. Si no tienes una distribución en CDROM, simplemente bájate los discos de instalación y arranque (ftp://ftp.FreeBSD.ORG/pub/FreeBSD/4.0-RELEASE/floppies/boot.flp) a tu disco duro, asegurándote de que tu navegador *grabe* el fichero en lugar de *mostrarlo*.

   Nota: Estas imágenes solo se pueden usar con discos de 1,44Mb.

4. Haz el disco de instalación a partir del fichero imágen:

   - Si estás usando MS-DOS o Windows bájate el programa fdimage.exe (ftp://ftp.FreeBSD.ORG/pub/FreeBSD/tools/fdimage.exe) o cógelo de `tools\fdimage.exe` en el CDROM y ejecútalo de la siguiente manera:

     ```
     E:\> tools\fdimage
     floppies\boot.flp a:
     ```

     El programa *fdimage* formateará el disco `A:` y copiará la imágen `boot.flp` en é:l (asumiendo que estás en el nivel superior de la distribució de FreeBSD y que las imágenes están en el subdirectorio `floppies` , tal y como suele ser habitual).

   - Si estás instalando un sistema UNIX para crear las imágenes de disco:

     ```
     # dd if=boot.flp of=disk_device
     ```

     `disk_device` es el dispositivo `/dev` para la disquetera. En sistemas FreeBSD, éste es `/dev/rfd0` para el disco `A:` y `/dev/rfd1` para el disco `B:`.

5. Con el disco de instalación en el disco `A:`, rearranca tu sistema. Deberías obtener un prompt de arranque como este:

   ```
   >> FreeBSD BOOT ...
   Usage: [[[0:][wd](0,a)]/kernel][-abcCdhrsv]
   Use 1:sd(0,a)kernel to boot sd0 if it is BIOS drive 1
   Use ? for file list or press Enter for defaults
   Boot:
   ```

   Si *no* tecleas nada, FreeBSD arrancará automáticamente con su configuración por defecto, después de una pausa de 5 segundos. Cuando FreeBSD arranca, comprueba tu ordenador para determinar el hardware instalado. El resultado de estas comprobaciones es mostrado en la pantalla.

6. Cuando el proceso de arranque finaliza, el menú principal de instalación de FreeBSD se muestra en pantalla.

Si algo va mal...

Debido a limitaciones en la arquitectura de los PC's, es imposible para el programa de prueba de hardware ser 100 por 100 fiable. En el caso de que tu hardware sea identificado incorrectamente, o que el proceso de prueba cuelgue la máquina, primero mira la sección configuraciones soportadas en esta guía de instalación para asegurar que tu hardware es soportado por FreeBSD.

Si tu hardware está soportado, resetea el ordenador y cuando aparece el prompt `Boot:` teclea `-c`. Esto hará que FreeBSD entre en modo de configuración de hardware. El kernel de FreeBSD en el disco de intalación está configurado asumiendo que la mayorí de dispositivos hardware están configurados tal y como viene de fábrica en lo referente a IRQ's, direcciones de memoria, canales DMA, etc. Si tu hardware ha sido reconfigurado, necesitarás usar la opción `-c` en el arranque para indicarle a FreeBSD donde se encuentra cada cosa.

También es posible que la prueba de un dispositivo no existente provoque una correcta detección de un dispositivo que sí está presente. En este caso, la prueba para este driver conflictivo deberí ser desactivada.

> **Aviso** No desactives ningún dispositivo que necesitas durante la instalación, como la pantalla (`sc0`).

En el modo de configuración puedes:

- Listar los drivers de dispositivos presentes en el kernel.
- Desactivar los drivers de hardware no instalado en tu sistema.
- Cambiar la IRQ, DRQ y direcciones IO usadas por los drivers de dispositivo.

Mientras aparece el prompt `config>`, teclea `help` para más información sobre los comandos disponibles. Después de ajustar los parámetros del kernel con el hardware que tienes instalado, teclea `quit` en el prompt `config>` para continuar el arranque con la nueva instalación.

Una vez FreeBSD está instalado, los cambios hechos en el modo de configuración serán guardados permanentemente para que no tengas que reconfigurar el sistema cada vez que arranques. Aún así, es aconsejable que te crees un kernel personalizado para optimizar el sistema. Mira en Configuración del kernel para crear un kernel personalizado.

2.1. Configuraciones soportadas

Actualmente FreeBSD se ejecuta en una variedad de buses ISA, VLB, EISA y PCI basados en PC's, cubriendo desde un 386sx hasta Pentiums (piensa que 386sx no es recomendado). También se soportan configuraciones IDE o ESDI, diferentes controladoras SCSI, tarjetas de red y serie, etc.

Un mínimo de cuatro megabytes de RAM son requeridos para ejecutar FreeBSD. Para ejecutar el Sistema X Window, ocho megabytes de RAM es el mínimo recomendado.

A continuación te presentamos una lista de todas las controladores de disco y tarjetas de red que actualmente funcionan en FreeBSD. Otras configuraciones posiblemente funcionen también, pero no hemos tenido noticias de ello.

2.1.1. Controladoras de disco

- WD1003 (any generic MFM/RLL)
- WD1007 (any generic IDE/ESDI)
- IDE
- ATA
- Adaptec 1505 ISA SCSI controller
- Adaptec 152x series ISA SCSI controllers
- Adaptec 1535 ISA SCSI controllers
- Adaptec 154x series ISA SCSI controllers
- Adaptec 174x series EISA SCSI controller in standard and enhanced mode.
- Adaptec 274x/284x/2940/2940U/3940 (Narrow/Wide/Twin) series EISA/VLB/PCI SCSI controllers
- Adaptec AIC7850 on-board SCSI controllers
- Adaptec AIC-6360 based boards, which includes the AHA-152x and SoundBlaster SCSI cards.

  Nota: No puedes arrancar desde tarjeta SounBlaster ya que no tienen BIOS, la cual es necesaria para mapear el dispositivo de arranque en los vectores de Entrada/Salida. Son perfectamente usables para cintas externas, CDROM's, etc. Lo mismo se aplica a cualquier otra tarjeta basada en la AIC-6x60 sin ROM de arranque. Algunos sistemas tienen una ROM de arranque, la cual es mostrada de alguna manera cuando el sistema arranca. Revisa la configuración de tu sistema/placa para más detalles.

- Buslogic 545S & 545c

  Nota: Buslogic was formerly known as "Bustek".

- Buslogic 445S/445c VLB SCSI controller

- Buslogic 742A/747S/747c EISA SCSI controller.
- Buslogic 946c PCI SCSI controller
- Buslogic 956c PCI SCSI controller
- NCR 53C810/53C815/53C825/53C860/53C875 PCI SCSI controller.
- NCR5380/NCR53400 (“ProAudio Spectrum”) SCSI controller.
- DTC 3290 EISA SCSI controller in 1542 emulation mode.
- UltraStor 14F/24F/34F SCSI controllers.
- Seagate ST01/02 SCSI controllers.
- Future Domain 8xx/950 series SCSI controllers.
- WD7000 SCSI controllers.

Con todas las controladoras SCSI soportadas, se consigue ofrecer soporte completo para periféricos SCSI-I & SCSI-II, incluyendo discos, cintas, DAT, y lectores de CDROM.

Los siguientes tipos de CDROM son soportados actualmente:

- SoundBlaster SCSI and ProAudio Spectrum SCSI (`cd`)
- Mitsumi (all models) proprietary interface (`mcd`)
- Matsushita/Panasonic (Creative) CR-562/CR-563 proprietary interface (`matcd`)
- Sony proprietary interface (`scd`)
- ATAPI IDE interface (experimental and should be considered ALPHA quality!) (`wcd`)

2.1.2. Tarjetas de red

- Allied-Telesis AT1700 and RE2000 cards
- SMC Elite 16 WD8013 Ethernet interface, and most other WD8003E, WD8003EBT, WD8003W, WD8013W, WD8003S, WD8003SBT and WD8013EBT based clones. SMC Elite Ultra and 9432TX based cards are also supported.
- DEC EtherWORKS III NICs (DE203, DE204, and DE205)
- DEC EtherWORKS II NICs (DE200, DE201, DE202, and DE422)
- DEC DC21040/DC21041/DC21140 based NICs:
  - ASUS PCI-L101-TB

  - Accton ENI1203
  - Cogent EM960PCI
  - Compex CPXPCI/32C
  - D-Link DE-530
  - DEC DE435
  - Danpex EN-9400P3
  - JCIS Condor JC1260
  - Kingston KNE100TX
  - Linksys EtherPCI
  - Mylex LNP101
  - SMC EtherPower 10/100 (Model 9332)
  - SMC EtherPower (Model 8432)
  - SMC EtherPower (2)
  - Zynx ZX314
  - Zynx ZX342

- DEC FDDI (DEFPA/DEFEA) NICs
- Fujitsu FMV-181 and FMV-182
- Fujitsu MB86960A/MB86965A
- Intel EtherExpress
- Intel EtherExpress Pro/100B 100Mbit.
- Isolan AT 4141-0 (16 bit)
- Isolink 4110 (8 bit)
- Lucent WaveLAN wireless networking interface.
- Novell NE1000, NE2000, and NE2100 ethernet interface.
- 3Com 3C501 cards
- 3Com 3C503 Etherlink II
- 3Com 3c505 Etherlink/+
- 3Com 3C507 Etherlink 16/TP

- 3Com 3C509, 3C579, 3C589 (PCMCIA) Etherlink III
- 3Com 3C590, 3C595 Etherlink III
- 3Com 3C90x cards.
- HP PC Lan Plus (27247B and 27252A)
- Toshiba ethernet cards
- PCMCIA ethernet cards from IBM and National Semiconductor are also supported.

> **Nota:** Actualmente FreeBSD no soporta las características PnP (plug-n-play) de algunas tarjetas. Si tu tarjeta tiene Pnp y te está dando problemas, intenta desactivarle el PnP.

2.1.3. Dispositivos varios

- AST 4 port serial card using shared IRQ.
- ARNET 8 port serial card using shared IRQ.
- BOCA IOAT66 6 port serial card using shared IRQ.
- BOCA 2016 16 port serial card using shared IRQ.
- Cyclades Cyclom-y Serial Board.
- STB 4 port card using shared IRQ.
- SDL Communications Riscom/8 Serial Board.
- SDL Communications RISCom/N2 and N2pci sync serial cards.
- Digiboard Sync/570i high-speed sync serial card.
- Decision-Computer Intl. “Eight-Serial” 8 port serial cards using shared IRQ.
- Adlib, SoundBlaster, SoundBlaster Pro, ProAudioSpectrum, Gravis UltraSound, Gravis UltraSound MAX and Roland MPU-401 sound cards.
- Matrox Meteor video frame grabber.
- Creative Labs Video spigot frame grabber.
- Omnimedia Talisman frame grabber.
- Brooktree BT848 chip based frame grabbers.
- X-10 power controllers.

- PC joystick and speaker.

FreeBSD actualmente no soporta el bus microcanal de IBM (MCA).

2.2. Preparándonos para la instalación

Existen diferentes métodos de instalación de FreeBSD. La siguiente sección describe la preparación necesaria para cada tipo de instalación.

2.2.1. Antes de instalar desde CDROM

Si tu lector de CDROM no está soportado, por favor, pasa a la sección Preparación en MSDOS.

No hay mucho trabajo de preparación previo para realizar una instalación satisfactoria desde uno de los CDROM's de Walnut Creek (otras distribuciones en CDROM también es posible que funcionen pero no tenemos manera de asegurarlo ya que no hemos podido probar ninguno, y tampoco sabemos como están hechos). Puedes arrancar directamente el programa de instalación desde DOS usando el fichero `install.bat` o puedes hacer los discos de arranque ejecutando el programa `makeflp.bat`.

> **Nota:** Si estás ejecutando FreeBSD 2.1-RELEASE y tienes un CDROM IDE, usa los ficheros `inst_ide.bat` o `atapiflp.bat`.

Para acceder al interface más sencillo de todos (desde DOS), ejecuta el comando `view`. Aparecerá una utilidad DOS con un menú a través del cual puedes acceder a todas las opciones de instalación disponibles.

Si estás creando el disco de arranque desde un sistema UNIX, mira el principio de esta guía .

Una vez hayas arrancado desde DOS o floppy, deberís ser capaz de poder seleccionar el CDROM como el soporte de instalación en el menú Media y cargar la distribución completa desde el CDROM. No se requiere ningús soporte adicional para la instalación.

Después de instalar por completo el sistema y arrancar desde el disco duro, puedes montar el CDROM en cualquier momento tecleando: `mount /cdrom`

Antes de poder quitar el CDROM, es necesario teclear: `umount /cdrom`.

> **Nota:** Antes de ejecutar el programa de instalación asegúrate de tener el CDROM en el lector para que éste pueda ser detectado durante la fase de pruebas de hardware. Esto es necesario también si quieres que el CDROM sea añadido automáticamente a la configuración inicial del sistema.

Finalmente, si quieres que tus usuarios puedan instalar FreeBSD vía FTP directamente desde el CDROM de tu máquina, lo puedes hacer de manera muy sencilla. Una vez tienes la máquina completamente instalada, sólo tienes que andir la siguiente línea al fichero de passwords (usando el comando vipw):

```
ftp:*:99:99::0:0:FTP:/cdrom:/nonexistent
```

Cualquiera con conectividad en tu red (y permisos para acceder a ella) puede seleccionar ahora como soporte de la instalación el tipo FTP y teclear: **`ftp://`*`tu_maquina`*** después de seleccionar “Other” en el menú de servidores FTP.

2.2.2. Antes de instalar desde Floppy

Si debes instalar desde disquettes, bién por problemas de compatibilidad de hardware o por que te guste hacer las cosas de la manera más complicada, antes debes preparar algunos disquettes.

Necesitarás, como mínimo, tantos disquettes de 1.44MB o 1.2MB como espacio ocupe la distribución en el directorio bin. Si estás preparando estos disquettes bajo DOS, entonces ESTOS disquettes *deben* estar formateados con el comando FORMAT de MS-DOS. Si utilizas Windows, usa el comando format del administrador de ficheros.

NO te fíes de los discos preformateados. Formatéalos tú mismo, sólo para estar seguro. Muchos de los problemas reportados en el pasado por usuarios, han tenido que ver con disquettes mal formateados o defectuosos.

Si estás creando los disquettes desde una máquina FreeBSD, formatear los discos tampoco es mala idea. Puedes usar el comando `disklabel` y `newfs` para formatearlos e incluir en ellos el sistema de ficheros UFS. Para hacerlo, puedes seguir la siguiente secuencia de comandos:

```
# fdformat -f 1440 fd0.1440

# disklabel -w -r fd0.1440 floppy3
# newfs -t 2 -u 18 -l 1 -i 65536 /dev/rfd0
```

Nota: Usa `fd0.1200` y `floppy5` para discos de 5.25" (o 1.2MB).

Ahora puedes montar los disquettes y escribir en ellos como en cualquier otro sistema de ficheros.

Después de formatear los disquettes, necesitarás copiar los ficheros en ellos. Los ficheros de la distribución están convenientemente distribuidos para que quepan 5 de ellos en cada disquette de 1.44Mb. Utiliza todos los discos necesarios para incluir en ellos todas las distribuciones que quieras instalar. Cada distribución debe estar en un subdirectorio del floppy, por ejemplo: `a:\bin\bin.aa`, `a:\bin\bin.ab`, y así hasta completar la distribución.

Una vez llegues a la pantalla de selección del soporte de la instalación, selecciona “Floppy” , siendo preguntado por el resto de parámetros necesarios.

2.2.3. Antes de instalar desde una partición MS-DOS

Para realizar la instalación desde una partición MS-DOS, copia los ficheros de la distribución en un directorio llamado `C:\FREEBSD`. La estructura de directorios del CDROM tiene que ser parcialmente reproducida dentro de éste directorio, por lo que te aconsejamos usar el comando `xcopy`. Por ejemplo, para preparar una instalación mínima de FreeBSD:

```
C:\> MD C:\FREEBSD

C:\> XCOPY /S E:\BIN C:\FREEBSD\BIN\
C:\> XCOPY /S E:\MANPAGES C:\FREEBSD\MANPAGES\
```

Asumiendo que `C:` es donde tienes espacio libre y `E:` es la unidad de tu lector de CDROM.

Tienes que copiar cada distribución que quieras instalar desde MS-DOS, bajo `C:\FREEBSD` — la distribución BIN es sólo la mínima requerida.

2.2.4. Antes de instalar desde una cinta QIC/SCSI

Instalar desde una cinta es probablemente el método más sencillo, aparte de una instalación on-line, vía FTP o CDROM. El programa de instalación espera que los ficheros estén en formato tar en la cinta, por lo que sólo tienes que seleccionar la distribución a instalar y copiarla en una cinta en formato tar con un comando como:

```
# cd /freebsd/distdir
# tar cvf /dev/rwt0 dist1 ... dist2
```

Cuando ejecutas la instalación, deberís asegurarte de dejar suficiente espacio libre en algún directorio temporal (el cual podrás elegir), para que el programa de instalación pueda recuperar *todo* el contenido de la cinta. Dado al acceso no aleatorio de las cintas, este método exige un poco de espacio temporal. Necesitarás tanto espacio temporal como contenidos hayan en la cinta.

Nota: Es imprescindible que la cinta esté en el lector *antes* de arrancar con el disco de instalación.

2.2.5. Antes de instalar sobre una red

Puedes hacer instalaciones de red mediante 3 tipos de conexión:

Puerto serie

SLIP o PPP

Puerto paralelo

PLIP (cable laplink)

Ethernet

Tarjeta ethernet estandar (incluye algunas PCMCIA).

El soporte de SLIP es bastante primitivo, y limitado a conexiones punto a punto como una cable serie entre un portátil y otro ordenador. La conexión debe ser mediante un cable cruzado serie ya que la instalación SLIP no ofrece posibilidad de marcado telefónico; esta facilidad se ofrece mediante la utilidad PPP, la cual aconsejamos usar siempre que sea posible.

Si estás usando un módem, PPP es tu única opción. Asegúrate de tener la información de tu proveedor a mano ya que el proceso de instalación te la pedirá de manera inmediata. Necesitarás saber como llamar a tu proveedor usando “comando AT” específicos de tu módem por que el marcador PPP sólo ofrece un simple emulador de terminal. Si usas PAP o CHAP, necesitarás teclear los comandos `set authname` y `set authkey` antes de teclear el comando `term`. Pásate por el manual ppp y las secciones de las FAQ (../FAQ/userppp.html) para más información. Si tienes problemas, el log puede ser dirigido a la pantalla usando el comando `set log local ...`.

Si puedes disponer de una conexión punto a punto a otro sistema FreeBSD (2.0R o superior), deberías considerar la instalación sobre el puerto paralelo con un cable “laplink”. La velocidad es mucho mayor que la que podemos conseguir sobre una conexión serie (por encima de los 50k/seg), obteniendo una instalación más rápida.

Finalmente, para la instalación de red más rápida posible, una tarjeta ethernet es siempre una buena elección. FreeBSD soporta muchas de las tarjetas ethernet del mercado, una tabla de las tarjetas soportadas (y sus características requeridas) está disponible en la sección Hardware soportado. Si estás usando una de las tarjeta PCMCIA soportadas, asegúrate de tener la tarjeta insertada *antes* de encender el portátil. Desafortunadamente, actualmente FreeBSD no soporta la "inserción en caliente"de tarjetas PCMCIA durante el proceso de instalación.

También necesitarás saber tu dirección IP en la red, el valor de la máscara de tu clase de direcciones y el nombre de tu máquina. Tu administrador de sistemas puede darte todos los valores adecuados a la configuración de tu red. Si vas a acceder a otras máquinas por nombre en lugar de dirección IP, necesitarás la dirección de un servidor de nombres y posiblemente la dirección de un gateway (si usas PPP es la dirección IP de tu proveedor). Si no conoces las respuestas a la mayoría de estas preguntas, definitivamente debes hablar con tu administrador de sistemas *antes* de intentar éste tipo de instalación.

Una vez tienes una conexión de red de cualquier tipo funcionando, la instalación puede continuar sobre NFS o FTP.

2.2.5.1. Preparando la instalación sobre NFS

La instalación sobre NFS es muy sencilla: simplemente copia los ficheros de las distribuciones de FreeBSD que necesites en algún lugar del servidor, y apunta el soporte de instalación de NFS hacia él.

Si este servidor sólo soporta accesos a “puertos seleccionados” (como ocurre generalmente con las estaciones de trabajo SUN), necesitarás activar esta opción en el menú Options antes de proceder con la instalación.

Si tienes una conexión de red de poca calidad, con ratios de transferencia muy pobres, también deberías activar el flag apropiado en el menú Options.

Para que la instalación por NFS funcione, el servidor debe soportar el acceso a subdirectorios exportados, por ejemplo, si el directorio de tu distribución de FreeBSD 4.0 reside en: `ziggy:/usr/archive/stuff/FreeBSD` Entonces `ziggy` debe permitir montar directamete `/usr/archive/stuff/FreeBSD`, y no sólo `/usr` o `/usr/archive/stuff`.

En fichero `/etc/exports` de FreeBSD, esto es controlado por la opción `-alldirs` . Otros servidores NFS pueden usar opciones diferentes. Si obtienes el mensaje Permission Denied por parte del servidor, significa que no tienes esta opción activada de manera adecuada.

2.2.5.2. Preparando una instalación por FTP

La instalación por FTP se puede realizar desde cualquiera de los servidores mirror que contengan una versión razonablemente actualizada de FreeBSD 4.0. Un completo menú de elecciones razonables para prácticamente cualquier parte del mundo está disponible en el menú de instalación FTP.

Si estás instalando desde cualquier otro servidor no listado en ese menú, o tienes problemas con la configuración del servidor de nombres o la resolución correcta, puedes especificar tu propia URL seleccionando la opción “Other” en el menú. Una URL también puede ser una dirección IP, de manera que lo siguiente funcionaría en la ausencia de un servidor de nombres:

```
ftp://165.113.121.81/pub/FreeBSD/4.0;-RELEASE
```

Hay dos modos de instalación FTP que puedes usar:

FTP Activo

Para todas las transferencias FTP, usa el modo “Activo”. Esto no funcionará a través de firewalls, pero funcionará sin problemas con servidores FTP antiguos que no soportan el modo pasivo. Si tu conexión se cuelga con el modo pasivo, intenta con el modo activo!.

FTP Pasivo

Para todas las transferencias FTP usa el modo “Pasivo”. Esto permite al usuario conectar a través de firewalls que no aceptan conexiones entrantes en direcciones de puertos aleatorias.

> **Nota:** Los modos Activos y Pasivos no son lo mismo que una conexión a través de “proxy”, donde un servidor proxy FTP está escuchando y reenviando las peticiones FTP!.

Para un servidor proxy FTP, normalmente deberías darle el nombre del servidor real al que quieres conectar como parte del nombre de usuario después de una @-sign. El servidor proxy reenviará la petición al servidor adecuado. Un ejemplo: digamos que quieres realizar la instalación desde `ftp.freebsd.org` usando el servidor proxy FTP `foo.bar.com`, que está escuchando en el puerto 1234.

En este caso, tienes que ir al menú de opciones, y poner el nombre de usuario de FTP como ftp@ftp.freebsd.org, y tu dirección de correo como password. Como medio de instalación , especifica FTP (o FTP pasivo, si el proxy lo soporta) y la URL `ftp://foo.bar.com:1234/pub/FreeBSD`

`/pub/FreeBSD` de `ftp.freebsd.org` es "cacheado"bajo `foo.bar.com`, permitiendote realizar la instalación desde *esa* máquina (la cual coge los ficheros de `ftp.freebsd.org` a medida que la instalación los necesita).

2.3. Instalando FreeBSD

Una vez has tomado nota de los pasos apropiados en la preinstalación, deberís poder instalar FreeBSD sin ningún tipo de problema.

Si tienes algún problema, deberías vover atrás y releer la sección de preparación aplicable a tu medio de instalación. Si estás teniendo problemas de hardware, o FreeBSD se niega a arrancar, lee la Guía de Hardware que puedes encontrar en el disco de arranque para posibles soluciones.

El disco de arranque de FreeBSD contiene toda la documentación on-line que deberís necesitar para realizar la instalación. Si no es así, nos gustaría saber que es lo que te ha confundido o dónde has tenido más problemas. Enví tus comentarios a Lista del Proyecto de Documentación de FreeBSD <`freebsd-doc@FreeBSD.ORG`>. El objetivo del programa de instalación de FreeBSD (sysinstall) es estar suficientemente documentado para poder realizar una instalación sin problemas.

Mientras tanto, puedes encontrar útil esta “típica secuencia de instalación”:

1. Arranca con el disco de arranque. Después de la secuencia de arranque que puede tardar entre 30 segundos y 3 minutos, dependiendo del hardware, debería aparecer un menú de opciones iniciales. Si el disco no arranca o el arranque se cuelga en algún momento, lee la sección Q&A de la Guía de Hardware para conocer las posibles causas.
2. Pulsa F1. Deberías ver unas instrucciones básicas sobre el sistema de menús y del programa en general. Si no has usado este sistema de menú anteriormente, *por favor* léelo.

3. Selecciona la opción Options y configura las preferencias que puedas tener o necesitas.
4. Selecciona la instalación Novice, Custom o Express, dependiendo de la ayuda que quieras recibir del programa de instalación durante todo el proceso. Si nunca has usado FreeBSD anteriormente, la instalación Novice es la más recomendada.
5. El menú de configuración final te permite configurar tu instalación de FreeBSD a través de un sistema guiado por menús. Algunas secciones, como la de red, pueden ser importantes si has hecho la instalación desde CDROM/Cinta/Floppy y no has configurado todavía tus interfaces de red (asumiendo que exista alguno). Configurando adecuadamente en este momento los interfaces permitiráa que FreeBSD esté activo en la red cuando reinicies por primera vez desde el disco duro.

2.4. Preguntas y Respuestas para usuarios de MS-DOS

Muchos usuarios de FreeBSD querrán instalar FreeBSD en PC's habitados por MS-DOS. Aquí están algunas de las pregunas (y respuestas) más comunes sobre la instalación de FreeBSD en estos sistemas.

Ayuda! No tengo espacio. Necesito borrarlo todo?

Si tu máquina está todavía ejecutando MS-DOS y tienes poco o ningún espacio libre disponible para la instalación de FreeBSD, no todo está perdido!. Quizás encuentres útil la utilidad FIPS, disponible en el directorio `tools` del CDROM de FreeBSD, o en cualquiera de los mirrors de FreeBSD.

FIPS te permite repartir una partición MS-DOS en dos partes, preservando la partición original permitiendo la instalación en la segunda. Primero debes defragmentar tu partición MS-DOS usando el programa DEFRAG de DOS 6.x o las utilidades Norton. Después ejecuta FIPS. El propio programa te preguntará todos los datos que necesite conocer. Finalmente puedes reiniciar la máquina e instalar FreeBSD en la segunda partición. Mira el menú *Distributions* para hacer una estimación de la cantidad de espacio necesario para el tipo de instalación que deseas hacer.

Puedo usar sistemas de ficheros comprimidos desde FreeBSD?

No. Si estás usando una utilidad como Stacker(tm) o DoubleSpace(tm), FreeBSD solo será capaz de usar la parte del sistema de ficheros no comprimida. El resto del sistema de ficheros se mostrarás como un único fichero. *No borres este fichero*. Lo agradecerás enormemente!.

Es mejor solución crear otra partición primaria de MS-DOS no comprimida y usarla para la comunicación entre MS-DOS y FreeBSD.

Puedo montar mis particiones MS-DOS extendidad?

Sí. Las particiones extendidas de DOS son mapeadas al final del otro "slices" en FreeBSD, por ejemplo, tu disco `D:` será `/dev/sd0s5`, tu disco `E:` será `/dev/sd0s6` y así respectivamente. Este ejemplo asume, por supuesto, que tu partición extendida está en el disco SCSI 0. Para discos IDE, sustituye `wd` por `sd`. Para montar las particiones extendidas tienes que hacerlo exactamente igual que cualquier otra partición o disco DOS:

```
# mount -t msdos /dev/sd0s5 /dos_d
```

Puedo ejecutar binarios MS-DOS bajo FreeBSD?

BSDI ha donado su emulador DOS al mundo BSD y éste ha sido portado a FreeBSD.

Hay también una (tecnicamente) buena aplicación disponible en la Colección de Ports llamada pcemu, la cual te permite ejecutar muchos binarios en modo texto emulando por completo una CPU 8088.

Capítulo 3. Unix básico

3.1. El manual en línea

La documentación más completa en FreeBSD estáa en forma de *páginas man*. Prácticamente todos los programas del sistema incluyen un pequeno manual de referencia explicando las operaciones básicas y los diferentes argumentos del programa. Estos manuales pueden verse con el comando `man`. El uso del comando `man` es simple:

```
% man command
```

`command` es el nombre del comando del que queremos aprender. Por ejemplo, para saber más sobre el comando `ls` teclea:

```
% man ls
```

El manual en línea está dividido en secciones numeradas:

1. Comandos de usuario
2. Llamadas de sistema y números de error
3. Funciones en las librerías de C
4. Drivers de dispositivos
5. Formatos de fichero
6. Juegos y otras diversiones
7. Información varia
8. Mantenimiento del sistema y comandos de sistema

En algunos casos, el mismo concepto aparece en más de una sección del manual en línea. Por ejemplo, hay un comando de usuario `chmod` y una llamada de sistema `chmod()`. En este caso, puedes espicificar al comando `man` cual de ellos quieres ver, especificando la sección:

```
% man 1 chmod
```

Esto mostrará la información del comando de usuario `chmod`. Las referencias a secciones particulares del manual en línea tradicionalmente se incluyen entre paréntesis en la documentación escrita, de manera que chmod(1) se refiere al comando de usuario `chmod` y chmod(2) se refiere a la llamada de sistema.

Esto es correcto si sabes el nombre del comando y simplemente quieres saber como usarlo, pero ¿qué pasa si no recuerdas el nombre del comando?. Puedes usar `man` para buscar palabras en las *descripciones* de los comandos usando el parámetro `-k`:

```
% man -k mail
```

Con este comando obtendrás una lista de todos los comandos que contienen en su descripción la palabra “mail”. Actualmente, este comando realiza la misma función que el comando `apropos`.

¿Quieres saber que hacen todos los comandos existentes en `/usr/bin`?. Simplemente haz:

```
% cd /usr/bin; man -f *
```

o

```
% cd /usr/bin; whatis *
```

ya que realizan la misma función .

3.2. Fichero GNU Info

FreeBSD incluye muchas aplicaciones y utilidades producidas por la Free Software Foundation (FSF). Como complemento a las páginas man, estos programas incluyen unos documentos hipertexto más extensos llamados ficheros “info” los cuales pueden visualizarse con el comando `info`, o, si tienes instalado `emacs`, con el modo info de `emacs`.

Para usar el comando info(1), simplemente teclea:

```
% info
```

Para una breve descripción, teclea **h**. Para una referencia rápida de comandos, teclea **?**.

Capítulo 4. Instalando Aplicaciones: La colección de Ports

Contribuido por James Raynard <jraynard@FreeBSD.org>.

La colección de Ports de FreeBSD te permite compilar e instalar una gran cantidad de programas con el mínimo esfuerzo.

Debido a las diferencias entre los estándars abiertos, conseguir que un programa funcione en una versión diferente de Unix puede ser tedioso y complicado, como debe saber todo aquel que lo haya intentado. Tendrás suerte si el programa que quieres compila limpiamente, instala todos los componentes donde debe instalarlos y funciona todo correctamente al primer intento.

Algunas distribuciones de software han intentado solucionar este problema incluyendo unos scripts de configuración. Algunos de estos scripts son muy inteligentes, pero tienen tendencia a anunciar que tu sistema es algo que nunca has oido y hacen preguntas que suenan a un exámen final de programación en Unix.

Afortunadamente, con la colección de Ports, todo el trabajo duro ya está hecho, y solo tienes que teclear el comando `make install` para tener un programa perfectamente instalado y en funcionamiento.

4.1. ¿Porqué tener una Colección de Ports?

El sistema base de FreeBSD incluye una gran variedad de herramientas y utilidades de sistema, pero muchos de los programas populares no están en la distribución base del sistema, por buenas razones:

1. Programas con los que algunos usuarios no puede vivir sin ellos y otros usuarios ni los conocen, como cierto editor basado en Lisp.
2. Programas demasiado especializados para ser incluidos en la distribución base del sistema (CAD, bases de datos, etc).
3. Programas que se pueden incluir en la categoría “Tengo que mirarlo cuando tenga un rato libre” (algunos lenguajes, por ejemplo).
4. Programas que son demasiado divertidos para ser incluidos en un sistema operativo serio como FreeBSD ;-)
5. Por muchos programas que se incluyesen en el sistema base, la gente siempre quiere más, y se debe crear una línea de separación en algún momento (de otra manera, las distribuciones de FreeBSD serín enormes).

Obviamente no sería razonable que cada usuario se portase manualmente sus programas favoritos (sin mencionar la enorme cantidad de trabajo duplicado), así que el proyecto FreeBSD ha usado un ingenioso sistema que mediante herramientas estándar permite automatizar todo el proceso.

Esta es una excelente ilustración de como el “Unix way” trabaja en la prática combinando una serie de simples pero muy flexibles herramientas y consiguiendo algo muy potente.

4.2. ¿Cómo funciona la Colección de Ports?

Los programas en Internet se suelen distribuir como un tarball consistente en un Makefile y el código fuente del programa, incluyendo algunas instrucciones (las cuales no siempre son tan instructivas como debieras), y un script de configuración.

El escenario habitual es que bajas el tarball vía FTP, lo extraes en algún directorio, lees las instrucciones, haces los cambios que parezcan necesarios, ejecutas el script de configuración y usas el programa `make` para compilar e instalar el programa desde el código fuente.

Los ports de FreeBSD siguen usando el mecanismo del tarball, pero usan un esqueleto en el que guardan la información necesaria para que el programa funcione correctamente en FreeBSD. Los ports también tienen su propio y personalizado Makefile para practicamente todos los ports puedan ser compilados de la misma manera.

Si miras el esqueleto de un port (ya sea en tu sistema FreeBSD (file://localhost/usr/ports/devel/ElectricFence) o en el servidor FTP (ftp://ftp.freebsd.org/pub/FreeBSD/ports/ports/devel/ElectricFence)) y esperas encontrar todo tipo de combinaciones de ciencia avanzada, puede que te decepciones, ya que sólo encontrarás uno o dos ficheros y directorios de lo más habitual. (En un momento veremos como obtener un port).

“¿Cómo es posible que esto pueda hacer algo? ” Oigo como lloras. “No hay código fuente!”

Lo creas o no, gentil lector, todo quedará entendido (o eso espero). Veamos que pasa si intentamos instalar un port. He elegido el programa **ElectricFence**, una útil herramienta para desarrolladores, ya que el esqueleto es más claro que muchos otros.

> **Nota:** Si intentas ejecutar esto, necesitas estar como root.

```
# cd /usr/ports/devel/ElectricFence
# make install
>> Checksum OK for ElectricFence-2.0.5.tar.gz.
===>  Extracting for ElectricFence-2.0.5
===>  Patching for ElectricFence-2.0.5
===>  Applying FreeBSD patches for ElectricFence-2.0.5
===>  Configuring for ElectricFence-2.0.5
===>  Building for ElectricFence-2.0.5
[lots of compiler output...]
===>  Installing for ElectricFence-2.0.5
===>  Warning: your umask is "0002". If this is not desired, set it to
      an appropriate value and install this port again by “make reinstall”.
```

```
install -c -o bin -g bin -m 444 /usr/ports/devel/ElectricFence/work/ElectricFence-2.0.5/libefenc
install -c -o bin -g bin -m 444 /usr/ports/devel/ElectricFence/work/ElectricFence-2.0.5/libefenc
===>  Compressing manual pages for ElectricFence-2.0.5
===>  Registering installation for ElectricFence-2.0.5
```

Para evitar confusiones, he borrado todas las líneas correspondientes a la compilación del programa.

Si has realizado la instalación de este port, habrás obtenido algo así en pantalla:

```
# make install
>> ElectricFence-2.0.5.tar.gz doesn't seem to exist on this system.
>> Attempting to fetch from ftp://ftp.doc.ic.ac.uk/Mirrors/sunsite.unc.edu/pub/Linux/devel/lang/
```

El programa `make` notifica que no tienes una copia en local del código fuente e intenta bajarlo por FTP. En el ejemplo, ya tenía el código en el ordenador, por lo que no ha sido necesario obtenerlo de Internet.

Vayamos revisando que ha hecho el programa `make`

1. Localizar el tarball con el código fuente. Si no está disponible localmente, intenta obtenerlo desde un servidor FTP.
2. Ejecutar un test de checksum para asegurar que el tarball del código fuente es correcto.
3. Extraer el tarball en un directorio temporal.
4. Aplicar todos los parches necesarios para que el código fuente compile y funcione bajo FreeBSD.
5. Ejecutar cualquier script de configuración requerido por el proceso de compillación, además de responder correctamente a cualquiera de las preguntas realizadas por éste.
6. (Finalmente!) Compilar el código.
7. Instalar los ejecutables del programa y otros archivos de soporte, páginas man, etc, bajo la jerarquía `/usr/local`, donde no se mezclarán con los programas de sistema. Esto también asegura que todos los ports que instales estarán en el mismo lugar, evitando que queden repartidos por diferentes lugares del disco.
8. Registrar la instalación en una base de datos. Esto significa que, si no te gusta el programa, puedes borrar todos sus programas y archivos de tu sistema.

Vuelve a mirar la salida del programa `make` antes mostrada e intenta localizar los pasos descritos. Y si no estabas impresionado antes, deberías estarlo ahora.

4.3. Obteniendo un Port de FreeBSD

Hay dos maneras de obtener un port de FreeBSD para un programa. Uno requiere el CDROM de FreeBSD, y el otro requiere una conexión a Internet.

4.3.1. Compilar los PORTS desde CDROM

Asumiendo que tu CDROM de FreeBSD está en el lector y montado en `/cdrom` (y *must* estar montado en `/cdrom`), deberís poder compilar toda la colección de ports sin problemas, ya que ésta encontraría los tarballs en `/cdrom/ports/distfiles/` (si existen allí) en lugar de obtenerlos de Internet.

Otra manera de hacer esto, si sólo quieres usar los esqueletos del CDROM, es poner la variables del fichero `/etc/make.conf` de la siguiente manera:

```
PORTSDIR=       /cdrom/ports
DISTDIR=        /tmp/distfiles
WRKDIRPREFIX=   /tmp
```

Substituye `/tmp` por cualquier directorio en el que tengas suficiente espacio. Entonces, entra en el subdirectorio apropiado bajo `/cdrom/ports` y teclea `make install` como hasta ahora. `WRKDIRPREFIX` hará que el port sea compilado bajo `/tmp/cdrom/ports`; por ejemplo, `games/oneko` será compilado bajo `/tmp/cdrom/ports/games/oneko`.

> **Nota:** Hay algunos ports para los que no podemos dar el código fuente original en el CDROM debido a limitaciones de licencia. Es estos casos, tendrás que mirar en la sección Compilando los ports usando una conexión a Internet.

4.3.2. Compilando los Ports desde Internet

Si no tienes CDROM o quieres estar seguro de instalar la última versión del port que te interesa, necesitarás bajarte el esqueleto de ese port.

Primero, si estás usando una versión release de FreeBSD, asegúrate de tener instalado el kit de actualización apropiado para tu release. Para conocer el kit apropiado, mira en la página web de ports (http://www.freebsd.org/ports/). Estos packages incluyen ficheros que han sido actualizados desde la release y que serán necesarios para compilar los nuevos ports.

La clave para los esqueletos es que el servidor FTP de FreeBSD puede crear tarballs al momento. Aquí tienes cómo trabaja, usando como ejemplo el programa gnats en el directorio de bases de datos (el texto entre corchetes son comentarios. No los teclees si intentas ejecutar el ejemplo):

```
# cd /usr/ports
# mkdir databases
# cd databases
# ftp ftp.freebsd.org
[log in as ‘ftp’ and give your email address when asked for a
password. Remember to use binary (also known as image) mode!]
```

```
> cd /pub/FreeBSD/ports/ports/databases
> get gnats.tar
[tars up the gnats skeleton for us]
> quit
# tar xf gnats.tar
[extract the gnats skeleton]
# cd gnats
# make install
[build and install gnats]
```

¿Qué ha ocurrido aquí?. Hemos conectado con el servidor FTP de la manera habitual y entrado en el subdirectorio `databases`. Al enviar el comando `get gnats.tar`, el servidor FTP ha empaquetado en formato tarred el directorio gnats.

A continuación hemos extraido el esqueleto de gnats y entrado en el directorio para compilar el port. Como hemos explicado anteriormente, el programa `make` ha detectado que el código fuente no estaba disponible localmente y lo ha bajado de Internet antes de extraerlo, aplicar los parches necesarios, compilarlo e instalarlo.

Intentemos ahora algo más ambicioso. En lugar de obtener el esqueleto de un simple port, obtengamos todos los de un directorio, por ejemplo los esqueletos de todas las bases de datos de la colección de ports. El proceso es muy parecido:

```
# cd /usr/ports
# ftp ftp.freebsd.org
[log in as ‘ftp’ and give your email address when asked for a
password. Remember to use binary (also known as image) mode!]
> cd /pub/FreeBSD/ports/ports
> get databases.tar
[tars up the databases directory for us]
> quit
# tar xf databases.tar
[extract all the database skeletons]
# cd databases
# make install
[build and install all the database ports]
```

Con media docena de sencillos comandos, tenenemos instalada toda una serie de programas de bases de datos en nuestra máquina FreeBSD. Todo lo que hemos hecho diferente de la instalación de un sólo port ha sido bajarnos todo un directorio y compilarlo todo de una sola vez. Impresionante, ¿no?.

Si tienes pensado instalar muchos ports, es buena idea bajarse la colección de ports completa.

4.4. Esqueletos

Un grupo de hackers compulsivos que ha olvidado comer en un intento de llegar a un punto prefijado?. No, un esqueleto aquí es la expresión mínima que incluye todo lo necesario para que los ports realicen su mágico trabajo.

4.4.1. `Makefile`

El componente más importante del esqueleto es el Makefile. Este contiene diferentes declaraciones que especifican cómo debe ser compilado e instalado un port. Aquí está el Makefile para el port ElectricFence:

```
# New ports collection makefile for:  Electric Fence
# Version required:     2.0.5
# Date created:         13 November 1997
# Whom:                 jraynard
#
# $Id$
#

DISTNAME=       ElectricFence-2.0.5
CATEGORIES=     devel
MASTER_SITES=   ${MASTER_SITE_SUNSITE}
MASTER_SITE_SUBDIR=     devel/lang/c

MAINTAINER=     jraynard@freebsd.org

MAN3=           libefence.3

do-install:
        ${INSTALL_DATA} ${WRKSRC}/libefence.a ${PREFIX}/lib
        ${INSTALL_MAN} ${WRKSRC}/libefence.3 ${PREFIX}/man/man3

.include <bsd.port.mk>
```

Las líneas que empiezan con el símbolo "#" son comentarios para facilitar las cosas a los lectores humanos (como en la mayoría de los scripts en Unix).

`DISTNAME` especifica el nombre del tarball, pero sin la extensión.

`CATEGORIES` indica que tipo de programa es. En este caso, una utilidad para desarrolladores. Mira en la sección categorías de este handbook para ver una lista completa.

`MASTER_SITES` es la URL(s) del servidor FTP principal, usado para obtener el tarball si no está disponible en el sistema local. Este servidor se considera fiable, y normalmente es desde el que se distribuye de manera oficial el programa.

`MAINTAINER` es la dirección de email de la persona responsable de actualizar el esqueleto, si, por ejemplo, aparece una nueva versión del programa.

Pasando por alto las siguientes líneas, la línea `.include <bsd.port.mk>` indica que las otras declaraciones o comandos necesarios para compilar el port están en un fichero estándar llamado `bsd.port.mk`. Como éstas son las mismas para todos los ports, no hay necesidad de duplicarlos en todos los ports, así que se mantienen en un solo fichero.

Este no es el lugar para entrar en detalle sobre el funcionamiento del Makefile; es suficiente con decir que la línea que comienza con `MAN3` asegura que la página man de ElectricFence sea compilada después de la instalación, para ayudar a conservar tu preciado espacio en disco. El port original no contenía el objeto `install`, así que las tres líneas a partir de `do-install` aseguran que los ficheros producidos por este port sean instalados en el lugar correcto.

4.4.2. El directorio `files`

El fichero que contiene el checksum del port se llama `md5`, ya que se usa el algoritmo MD5 para comprobar el checksum de los ports. Está en el directorio llamado `files`.

Este directorio puede contener otros ficheros necesarios para compilar el port y que no pueden situarse en ningun otro lugar.

4.4.3. El directorio `patches`

Este directorio contiene los patches necesarios para hacer que todo funcione correctamente bajo FreeBSD.

4.4.4. El directorio `pkg`

Este programa contiene tres archivos muy comunes:

- `COMMENT` — una descripción muy corta del programa.
- `DESCR` — una descripción más detallada.
- `PLIST` — una lista de todos los archivos que serán creados cuando el programa esté instalado.

4.5. ¿Qué hacer cuando un port no funciona?

Oh, puedes hacer una de estas cuatro (4) cosas:

1. Solucionarlo tú mismo. Los detalles técnicos de como trabajan los ports se pueden encontrar en Portando aplicaciones.
2. Quejarte. *Sólo* hacerlo por email. Enví el mail a la dirección Lista de Ports de FreeBSD <freebsd-ports@FreeBSD.ORG> y por favor, indica el nombre/versión del port, dónde obtuviste el código fuente y cual ha sido el texto de error.
3. Olvidarlo. Este el sistema más fácil para alguno de los pocos ports que puedan clasificarse como esenciales.
4. Obtener el package compilado desde un servidor FTP. La colección principal de packages está en el servidor principal de FreeBSD en el directorio packages (ftp://ftp.FreeBSD.org/pub/FreeBSD/ports/packages/), pero por favor, mira primero en tu mirror local. Estos programas suelen dar menos problemas que intentar compilar el código fuente y es, sobre todo, mucho más rápido. Usa el programa pkg_add(1) para instalar un package en tu sistema.

4.6. Algunas Preguntas y Respuestas

- P. Pensaba que esto iba a ser una discusión sobre modems

  R. Ah. Debes estar pensando en los puertos serie que hay en la parte trasera de tu ordenador. Aquí usamos la palabra “port” para definir el resultado de “portar” un programa de una versión de Unix a otra. (Desafortunadamente es un mal hábito de los informáticos el usar la misma palabra para referirse a cosas completamente diferentes).

- P. Pensaba que usabais los packages para instalar los programas extras

  R. Sí, esa es la manera más rápida y fácil de instalarlos.

- P. Entonces, ¿Porqué usar los ports?

  R. Por diferentes razones:

  1. Las condiciones de la licencia de algunos programas requieren que sean distribuidos como código fuente y no en formato binario.
  2. Algunas personas no confían en las distribuciones binarias. Al menos, con el código fuente puedes (en teoría) repasarlo y detectar problemas potenciales tú mismo.
  3. Si tienes algunos parches locales, necesitas el código fuente para poder añadirlos tú mismo.

4. Posiblemente tengas opiniones diferentes a las de otros usuarios en lo que respecta a las opciones de optimización en la compilación, creación de versiones de debug, etc.
5. A algunos usuarios les gusta tener el código fuente para poder leerlo, retocarlo, destrozarlo (dentro de los términos de la licencia, claro), y cosas así.

- P. ¿Qué es un patch (parche)?

  R. Un parche es un pequeno fichero (normalmente) que especifica como pasar de una versión de un archivo a otra. Contiene texto que especifica cosas como “borrar línea 23”, “anadir estas dos líneas después de la línea 468” o “cambia la línea 197 por esta”. También se conoce como “diff” ya que se genera por un programa llamado así.

- P. Qué es un tarball?

  R. Es un archivo terminado en `.tar` o `.tar.gz` (con variaciones como `.tar.Z`, o también `.tgz` si intentas adecuar el nombre del archivo a un sistema de archivos DOS).

  Básicamente, es un árbol de directorios que ha sido archivado en un solo fichero (`.tar`) y opcionalmente comprimido (`.gz`). Esta técnica se usó originalmente para *T*ape *AR*chives (archivos de cinta) y de aquí el nombre `tar`, pero ahora es de uso general en la distribución deil código fuente de programas a través de Internet.

  Puedes ver que archivos hay en ellos, o bién extraerlos tú mismo, usando el programa estándar de Unix tar, incluido en el sistema base de FreeBSD, así:

  ```
  % tar tvzf foobar.tar.gz

  % tar xzvf foobar.tar.gz
  % tar tvf foobar.tar
  % tar xvf foobar.tar
  ```

- P. ¿Y el checksum?

  R. Es un número que se genera a partir de todo el contenido del archivo que quieres chequear. Si cambia alguno de los carácteres, el checksum no será el mismo, así que una simple comparación te permitirá conocer la diferencia.

- P. He hecho lo que indicais para compilar los ports desde el CDROM y todo ha funcionado bién hasta que he intentado instalar el port kermit:

  ```
  # make install
  >> cku190.tar.gz doesn’t seem to exist on this system.
  >> Attempting to fetch from ftp://kermit.columbia.edu/kermit/archives/.
  ```

  ¿Porqué no puedo encontrarlo?

R. Los términos de la licencia del programa kermit no nos permiten incluir el tarball en el CDROM, así que tendrás que bajarlo a mano. La razón por la que han salido esos errores es que no estabas conectado a Internet en el momento de la instalación. Una vez lo hayas bajado de cualquiera de los servidores arriba mencionados, puedes empezar de nuevo el proceso (intenta escoger el servidor más cercano a tí para ganar tiempo y ahorrar ancho de banda de Internet).

- P. He hecho lo que explicais, pero cuando intento poner el archivo en `/usr/ports/distfiles` aparece un error que indica que no tengo permisos.

  R. El mecanismo de ports busca el tarball en `/usr/ports/distfiles`, pero no podrás copiar nada ahí por que es un link al CDROM, el cual es de sólo lectura. Puedes indicar que busque el port en cualquier otro directorio haciendo

  ```
  # make DISTDIR=/where/you/put/it install
  ```

- P. ¿Sólo funciona el esquema de ports si están en `/usr/ports`? Mi administrador de sistema dice que tengo que poner los archivos en `/u/people/guests/wurzburger`, pero parece que así no funciona.

  R. Puedes usar las variables `PORTSDIR` y `PREFIX` para indicarle al mecanismo de ports que use directorios diferentes. Por ejemplo,

  ```
  # make PORTSDIR=/u/people/guests/wurzburger/ports install
  ```

  compilará el port en `/u/people/guests/wurzburger/ports` e instalará todos los archivos bajo `/usr/local`.

  ```
  # make PREFIX=/u/people/guests/wurzburger/local install
  ```

  compilará el port en `/usr/ports` y lo instalará en `/u/people/guests/wurzburger/local`.

  Y por supuesto

  ```
  # make PORTSDIR=.../ports PREFIX=.../local install
  ```

  combinará los dos (es demasiado extenso para escribirlo en la página, pero seguro que te haces una idea).

  Si no quieres tener que teclear todo esto cada vez que instalas un ports, es una buena idea poner estas variables como variables de entorno.

- P. No tengo el CDROM de FreeBSD pero me gustaría disponer de todos los tarball en mi sistema para no tener que esperar a que se bajen de Internet cada vez que instalo un port. Hay alguna manera sencilla de hacerlo?

  R. Para obtener todos los tarball de la colección completa de ports haz

  ```
  # cd /usr/ports
  # make fetch
  ```

  Para todos los tarball de un directorio de ports, haz

```
# cd /usr/ports/directory
# make fetch
```

y para un solo port, bueno, creo que ya lo has adivinado.

- P. Sé que probablemente es más rápido obtener los tarballs desde un mirror de FreeBSD cercano. ¿Hay alguna manera de obtener los ports de otros servidores diferentes a los establecidos en MASTER_SITES?

  R. Sí. Si sabes, por ejemplo que `ftp.FreeBSD.ORG` te es más cercano que los servidores listados en `MASTER_SITES`, haz como en el siguiente ejemplo.

  ```
  # cd /usr/ports/directory
  # make MASTER_SITE_OVERRIDE=ftp://ftp.FreeBSD.ORG/pub/FreeBSD/ports/distfiles/ fetch
  ```

- P. Quiero saber que ficheros va a necesitar el programa `make` antes de que intente bajarlos.

  R. `make fetch-list` mostrará una lista de los ficheros necesarios para el port.

- P. ¿Hay alguna manera de parar la compilación del port?. Quiero hacer algunos retoques en el código fuente antes de instalarlo, pero es un poco pesado tener que estar atento y pulsar control-C cada vez.

  R. Ejecutando `make extract`, el port parará después de haber obtenido y extraido el código fuente.

- P. Estoy intentando crear mi propio port y quiero poder parar la compilación hasta poder asegurarme de que mis parches funcionan correctamente. ¿Hay alguna comando como `make extract`, pero para parches?

  R. Sí, `make patch` es lo que quieres. Probablemente también encontrarás muy útil la opción `PATCH_DEBUG`. Y ya de paso, gracias por el esfuerzo!.

- P. He oido que algunas opciones de compilación pueden causar errores. ¿Es cierto?. ¿Cómo puedo asegurarme de que compilo los ports con las opciones correctas?.

  R. Sí, con la versión 2.6.3 de `gcc` (la versión incluida en FreeBSD 2.1.0 y 2.1.5), la opción `-O2` puede crear problemas a no ser que uses también la opción `-fno-strength-reduce`. (Muchos de los ports no usan `-O2`). *Deberís* poder especificar las opciones de compilación con algo como

  ```
  # make CFLAGS='-O2 -fno-strength-reduce'install
  ```

  o editando el fichero `/etc/make.conf`, pero desafortunadamente no todos los ports respetan esto. La forma más segura es hacer `make configure`, entrar en el directorio de los fuentes e inspeccionar a mano los Makefiles, pero puede ser muy tedioso si los fuentes tienen muchos subdirectorios, cada uno con su propio Makefile.

- P. Hay tantos ports que resulta complicado encontrar el que quiero. ¿Hay alguna lista de los ports disponibles?

  R. Mira el archivo `INDEX` en `/usr/ports`. También puedes hacer búsquedas de palabras en la colección de ports. Por ejemplo puedes encontrar ports referentes al lenguaje de programación LISP usando:

  ```
  % cd /usr/ports
  % make search key=lisp
  ```

- P. Iba a instalar el port `foo` pero de pronto el sistema paró la compilación y comenzó a compilar el port `bar`. ¿Qué está pasando?

  R. El port `foo` necesita algo que ofrece el port `bar`, por ejemplo, si `foo` usa gráficos, `bar` puede tener librerías con rutinas de procesamiento gráfico útiles. O puede ser que `bar` sea una herramienta necesaria para compilar el port `foo`.

- P. Instalé el programa `grizzle` desde los ports, y, francamente, me parece una pérdida de espacio en disco, Quiero borrarlo pero no sé dónde están todos los ficheros. ¿alguna idea?

  R. No hay problema, sólo haz

  ```
  # pkg_delete grizzle-6.5
  ```

- P. Espera un momento; necesitas saber el número de versión para usar ese comando. No creerás que voy a ser capaz de recordarlo, ¿no?

  R. No, desde luego. Puedes encontrar el número de versión haciendo

  ```
  # pkg_info -a | grep grizzle
  Information for grizzle-6.5:
  grizzle-6.5 - the combined piano tutorial, LOGO interpreter and shoot 'em up arcade game.
  ```

- P. Hablando de espacio en disco, el directorio de ports parece que cada vez ocupa más espacio. ¿Es recomendable borrar cosas?

  R. Sí, si ya tienes instalado el programa y crees que no vas a necesitarlo de nuevo. La mejor manera de hacerlo es

  ```
  # cd /usr/ports
  # make clean
  ```

  lo que entrará en todos los subdirectorios de ports y borrará todo excepto el esqueleto de cada port.

- P. He hecho lo que comentais y todavía tengo todos esos tarballs o como le llameis en el directorio `distfiles`. ¿Puedo borrar esos archivos?

R. Sí, si estás seguro de haber terminado con ellos.

- P. Me gusta tener muchos programas para poder jugar con ellos. ¿Hay alguna manera de instalar todos los ports de una vez?

  R. Sólo haz

  ```
  # cd /usr/ports
  # make install
  ```

- P. Bién, lo he intentado, pero pensando que tardaría mucho me fuí a dormir y cuando he vuelto esta mañna he visto que sólo había instalado tres ports y medio. ¿Ha ido algo mal?

  R. No, el problema es que algunos ports necesitan hacer preguntas que no podemos responder por tí y necesitan tener a alguien "a mano"para poder responderlas.

- P. Realmente, no quiero perder todo un día delante del monitor. ¿Alguna idea mejor?

  R. Bueno, haz esto antes de irte a dormir

  ```
  # cd /usr/ports
  # make -DBATCH install
  ```

  Esto instalará todos los ports que *no* requieran participación por parte del usuario. Entonces, cuando vuelvas haz

  ```
  # cd /usr/ports
  # make -DIS_INTERACTIVE install
  ```

  para terminar el trabajo.

- P. En el trabajo, estamos usando el programa `frobble`, que está en la colección de ports, pero lo hemos alterado un poco para cubrir nuestras necesidades. ¿Hay alguna manera sencilla de hacer nuestros propios packages de manera que podamos distribuirlos más fácilmente en nuestros servidores?

  R. No hay problema, asumiendo que sabes como hacer los parches para tus cambios:

  ```
  # cd /usr/ports/somewhere/frobble
  # make extract
  # cd work/frobble-2.8
  [Apply your patches]
  # cd ../..
  # make package
  ```

- P. Este sistema de ports es realmente fantástici. Estoy desesperado por saber como lo habeis hecho. ¿Cuál es el secreto?

  R. No hay nada secreto, lo tienes todo en los ficheros `bsd.ports.mk` y `bsd.ports.subdir.mk` en tu directorio de makefiles. (file://localhost/usr/share/mk/)

  Nota: Los lectores con aversión a intrincados shell-scripts están avisados de no seguir este link...)

4.7. Haciendo tú mismo un port

Contribuido por Jordan K. Hubbard <`jkh@FreeBSD.org`>, Gary Palmer <`gpalmer@FreeBSD.org`>, Satoshi Asami <`asami@FreeBSD.org`> David O'Brien <`obrien@FreeBSD.org`> y Tim Vanderhoek <`hoek@FreeBSD.org`>. 28 de Agosto de 1996.

Así que, ¿ahora estás interesado en hacer un port?. Bién!

Lo que viene a continuación son algunas guís para crear un nuevo port para FreeBSD. La mayoría del trabajo es hecha por el fichero `/usr/ports/Mk/bsd.port.mk` que incluyen todos los Makefile de los ports. Por favor, mira en este fichero para más información sobre la manera de trabajar de la colección de ports. Aunque no uses los Makefiles de manera habitualm, está bién comentado, y podrás conocer una gran cantidad de cosas a través de él.

Nota: Sólo una pequeña parte de las variables sobreescribibles (*`VAR`*) se mencionan en este documento. Muchas (si no todas) están documentadas al inicio del fichero `bsd.port.mk`. Este fichero usa una tabulación no estándar. **Emacs** y **Vim** deberín reconocer la configuración al cargar el fichero. `vi` or `ex` se pueden configurar para que usen los valores correctos tecleando `:set tabstop=4` una vez que el fichero ha sido cargado.

4.7.1. Haciendo Ports rápidamente

Esta sección te explica cómo hacer un port rápidamente. En muchos casos no es suficiente, pero ya lo iremos viendo.

Primero, obtén el tarball original y ponlo en `DISTDIR`, que por defecto apunta a `/usr/ports/distfiles`.

Nota: Los siguientes comentarios asumen que el software compila sin problema alguno en FreeBSD, y no se requiere absolutamente ningún cambio para que el port funcione en tu sistema FreeBSD. Si has necesitado modificar alguna cosa, tendrás que referirte también por la siguiente sección.

4.7.1.1. Escribiendo el `Makefile`

El `Makefile` debería ser algo como esto:

```
# New ports collection makefile for:   oneko
# Version required:    1.1b
# Date created:        5 December 1994
# Whom:                asami
#
# $Id$
#

DISTNAME=      oneko-1.1b
CATEGORIES=    games
MASTER_SITES=  ftp://ftp.cs.columbia.edu/archives/X11R5/contrib/

MAINTAINER=    asami@FreeBSD.ORG

MAN1=          oneko.1
MANCOMPRESSED= yes
USE_IMAKE=     yes

.include <bsd.port.mk>
```

Prueba de entenderlo tú mismo. No te preocupes por los contenidos de la línea `$Id$`, ya que serán completadas automaticamente por el CVS cuando el port sea importado a la colección de ports principal. Puedes encontrar información más detallada en la sección Makefile de ejemplo.

4.7.1.2. Escribiendo los ficheros de descripción

Hay tres ficheros de descripción que son requeridos por cualquier port. Los ficheros son `COMMENT`, `DESCR` y `PLIST`. Estos ficheros deben estar en el subdirectorio `pkg`.

4.7.1.2.1. `COMMENT`

Descripción corta del port. *Por favor* no incluir el nombre del package (o número de versión del software) en el comentario. Aquí tienes un ejemplo:

```
A cat chasing a mouse all over the screen.
```

4.7.1.2.2. `DESCR`

Esta es una descripción más detallada del port. Uno o varios párrafos explicando claramente que hace el port es suficiente.

> **Nota:** Este *no* es un manual o una descripción en profundidad de cómo usar o compilar el port! *Por favorm, ten cuidado si lo estás copiando del `README` o del man*. Si el software portado tiene una página web oficial, deberías listarlo aquí.

Te recomendamos que pongas tu nombre o firma al final de este fichero de la manerae siguiente:

```
This is a port of oneko, in which a cat chases a poor mouse all over
the screen.
 :
(etc.)

http://www.oneko.org/

- Satoshi
asami@cs.berkeley.edu
```

4.7.1.2.3. `PLIST`

Este fichero lista todos los ficheros instalados por el port. También se llama “packing list” por que el package es generado con los ficheros listados aquí. Los paths son relativos al prefijo de instalación (normalmente `/usr/local` o `/usr/X11R6`). Si estás usando las variables `MAN`*n* (como deberís hacer), no listes ninguna página man aquí.

Aquí tienes un pequeno ejemplo:

```
bin/oneko
lib/X11/app-defaults/Oneko
lib/X11/oneko/cat1.xpm
lib/X11/oneko/cat2.xpm
lib/X11/oneko/mouse.xpm
@dirrm lib/X11/oneko
```

Mira en el man de pkg_create(1) para más detalles sobre la "lista de empaquetado".

> **Nota:** Deberís listar todos los ficheros, pero no los nombres de los directorios. De la misma manera, si el port crea directorios por sí mismo durante la instalación, asegúrate de añadir las líneas `@dirrm` cuando sea necesario para eliminarlos cuando el port sea desinstalado.
>
> Es recomendable que mantengas todos los nobres de fichero en este archivo ordenados alfabéticamente. Hará que la verificación de cambios cuando actualices el port sea mucho más sencilla.

4.7.1.3. Creando el fichero de checksum

Solo teclea `make checksum`. Las reglas del programa make para los ports generarán automáticamente el fichero `files/md5`.

4.7.1.4. Testeando los ports

Debes asegurarte de que las reglas de los ports hagan exactamente lo que quieres que hagan, incluyendo el empaquetado del port. Estos son los puntos importantes que necesitas verificar:

- `PLIST` no contiene nada que no sea instalado por tu port
- `PLIST` contiene absolutamente todos los ficheros instalados por tu port
- Tu port puede ser instalado múltiples veces usando el parámetro `reinstall`
- Tu port se elimina a sí mismo al usar el parámetro deinstall

Orden de test recomendado

1. `make install`
2. `make package`
3. `make deinstall`
4. `pkg_add ‘make package-name‘`
5. `make deinstall`
6. `make reinstall`
7. `make package`

Asegúrate de que no existe ningún warning en los comandos `package` y `deinstall`. Después del paso 3, mira si todos los nuevos directorios han sido eliminados correctamente. también, intenta usar el software después del paso 4, para asegurarte de que funciona correctamente cuando es instalado desde un package.

4.7.1.5. Probando tu port con `portlint`

Por favor, usa el comando `portlint` para ver si tu port cumple nuestras normas. El programa `portlint` forma parte de la colección de ports. En particular, debes probar que el Makefile es correcto y que el package es nombrado apropiadamente.

4.7.1.6. Enviando un port

Primero, asegúrate de haber leido la sección Do's and Dont's.

Ahora que ya tienes tu port, solo queda incluirlo en la colección de ports principal de FreeBSD y que todo el mundo lo tenga a su alcance. No necesitamos tu directorio `work` o el archivo de package `pkgname.tgz`, así que bórralos ahora. A continuación, simplemente incluye la salida de `shar 'find port_dir'` en un "bug report"y envialo con el programa send-pr(1) (mira en Bug Reports y comentarios generales para más información sobre send-pr(1). Si el port descomprimido es mayor de 20Kb deberías comprimirlo en un fichero tar y usar el comando uuencode(1) antes de incluirlo en el bug report. Clasifica el bug report en la categoría `ports` y clase `change-request`. No marques el port como `confidencial`!)

Una vez más, *no incluyas el código fuente original, el directorio `work`, o el package creado con `make package`*.

> **Nota:** En el pasado, te recomendabamos enviar el nuevo port a nuestro servidor ftp (`ftp.freebsd.org`). Esto ya no es recomendando ya que el permiso de lectura se ha desactivado en el directorio `incoming` por la cantidad de software pirata que se enviaba.

Miraremos tu port, nos pondremos en contacto contigo si es necesario y lo anadiremos en la colección principal. Tu nombre aparecerá en la lista "Additional FreeBSD contributors" en el Handbook de FreeBSD y otros ficheros. ¿No es genial?!? :)

4.7.2. Ports paso a paso

Bién, así que no ha sido tan simple, y el port ha necesitado algunas modificaciones para funcionar. En esta sección, explicaremos, paso a paso, como modificarlo para que pueda funcionar en el entorno de los ports.

4.7.2.1. Cócomo funcionan las cosas

Primero, esta es la secuencia de eventos que ocurren cuando un usuario teclea `make` en el directorio de ports. Si tienes otra ventana con el fichero `bsd.port.mk` te resultará más sencillo entender todo el proceso.

Pero no te preocupes si realmente no entiendes todo lo que hace el fichero `bsd.port.mk`... no hay mucha gente que realmente lo entienda... :>

1. El objeto `fetch` es ejecutado. El objeto `fetch` es el responsable de asegurar que el tarball necesario existe localmente en el directorio `DISTDIR`. Si `fetch` no puede encontrar los ficheros requeridos en `DISTDIR` localizará el URL de `MASTER_SITES`, la cual existe en el Makefile, al igual que en nuestro servidor principal ftp ftp://ftp.freebsd.org/pub/FreeBSD/ports/distfiles/, que usamos como servidor de backup. A continuación intentará obtener el archivo de la distribución con `FETCH`, asumiendo que el servidor de origen tiene conexión directa a Internet. Si este paso termina correctamente, grabará el archivo en `DISTDIR` para su uso futuro.
2. El objeto `extract` es ejecutado. Busca el archivo de distribución del port (normalmente un tarball comprimido con gzip) en `DISTDIR` y lo descomprime en un sibdirectorio temporal especificado por la variable `WRKDIR` (normalmente `work`).
3. Se ejecuta el objeto `patch`. Primero se aplican todos los parches definidos en `PATCHFILES`. A continuación, si no se han encontrado parches en `PATCHDIR` (que por defecto está definido como el subdirectorio `patches`), son aplicados en orden alfabético.
4. Se ejecuta el objeto `configure`. Este puede hacer cualquiera de las siguientes funciones.
    1. Si existe, se ejecuta `scripts/configure`.
    2. Si `USE_IMAKE` está definido, se ejecuta `XMKMF` (por defecto: `xmkmf -a`).
5. Se ejecuta el objeto `build`. Este es responsable de descender por los directorios de trabajo privados del port (`WRKSRC`) y compilarlos. Si se ha definido `USE_GMAKE` se usa el comando `make`, en caso contrario, se usará el comando de sistema `make`.

Las anteriores, son las acciones por defecto. Además, puedes definir los objetos `pre-`*`algo`* o `post-`*`something`*, o poner scripts con esos nombres. En el subdirectorio `scripts`, y serán ejecutados antes o después de las acciones por defecto.

Por ejemplo, si tienes definido un objeto `post-extract` en tu Makefile, y un fichero `pre-build` en el subdirectorio `scripts`, el objeto `post-extract` será llamado después de las acciones de extracción regulares, y el script `pre-build` será ejecutado antes que las reglas por defecto de compilación. Es recomendable que uses objetos de `Makefile` en caso de que las acciones a realizar sean suficientemente simples, ya que así resultará mucho más sencillo saber que tipo de acciones no regulares requiere el port.

Las acciones por defecto son ejecutadas por los objetos `do-`*`something`* del fichero `bsd.port.mk`. Por ejemplo, los mandatos para extraer un port están en el objeto `do-extract`. Si necesitas algo no incluido en el objeto por defecto, puedes solucionarlo redefiniendo el objeto `do-`*`something`* en tu `Makefile`.

> **Nota:** Los objetos “principales” (por ejemplo, `extract`, `configure`, etc.) no hacen nada más que asegurar que se han completado todos los procesos y llamar a los objetos o scripts reales, y no se espera que sean modificados. Si quieres modificar la extracción, cambia el objeto `do-extract`, pero nunca modifiques el objeto `extract`!

Ahora que entiendes lo que pasa cuando un usuario teclea `make`, déjanos explicarte los pasos recomendados para crear el port perfecto.

4.7.2.2. Obteniendo los fuentes originales

Obtén los fuentes originales (normalmente) en un archivo comprimido (*`foo`*`.tar.gz` o *`foo`*`.tar.Z`) y copialo en el directorio `DISTDIR`. Usa siempre dónde y cuando puedas los fuentes procedentes del servidor *principal* de la distribución.

Si no puedes encontrar un servidor ftp/http con una buena conexión a Internet, o sólo encuentras servidores que tienen los fuentes en irritantes formatos no estándar, puedes poner una copia funcional del archivo en un servidor ftp o http bajo tu control (tu página personal, por ejemplo). Asegúrate de usar el valor correcto en la variable `MASTER_SITES` para que refleje tu elección.

Si no puedes encontrar ningún lugar en el que poner el fichero de distribución (si eres committer de FreeBSD, puedes ponerlo en el directorio `public_html/` del servidor `freefall`), nosotros podemos albergarlo en `ftp://ftp.freebsd.org/pub/FreeBSD/ports/distfiles/LOCAL_PORTS/` como último recurso. Por favor, usa la variable `MASTER_SITE_LOCAL` para referirte a este servidor y subdirectorio. Envía un mail a Lista de Ports de FreeBSD <`freebsd-ports@FreeBSD.ORG`> si no estás seguro de qué hacer.

Si tu archivo de distribución cambia continuamente sin una buena razón, considera poner el archivo en tu página personal y listarlo como el primer `MASTER_SITES`. Esto evitará que los usuarios obtengan errores del tipo checksum mismatch`checksum mismatch`checksum mismatch, y también reducirá la carga de trabajo de las personas que hacen el mantenimiento de nuestro servidor FTP. De la misma manera, si existe un solo servidor como distribuidor principal del programa, es recomendable que albergues una copia de la distribución en tu servidor y lo listes como segundo `MASTER_SITES`.

Si tu port requiere algunos parches adicionales que están disponibles en Internet, bájalos y ponlos en `DISTDIR`. No te preocupes si provienen de un lugar diferente del que obtuviste el tarball principal, tenemos una manera de solucionar este problema (consulta la descripción de PATCHFILES).

4.7.2.3. Modificando el port

Descomprime una copia del tarball en un directorio privado y haz todos los cambios necesarios para que el port compile en la versión actual de FreeBSD. Mantén *una lista completa* de todas las modificaciones que realizas para poder automatizar el proceso. Todo, incluyendo borrar, anadir o modificar ficheros debe poder hacerse de manera automática usando un script o parche.

Si tu port requiere una importante interacción o personalización para compilar o instalar, puedes mirar la clásica aplicación **Configure** de Larry Wall y hacer algo así tu mismo. El objetivo de la nueva colección de ports es hacer que cada port sea tan “plug-and-play” como sea posible para el usuario final, usando el mínimo espacio en disco posible.

Nota: Aunque no se haya mencionado explícitamente, los parches, scripts y otros archivos que hayas creado o contribuido a la colección de ports de FreeBSD están cubiertos por las condiciones estándar del copyright BSD.

4.7.2.4. Parcheando

En la preparación del port, los archivos que han sido andidos o modificados pueden recorrerse con un diff recursivo. Cada serie de parches que quieras aplicar deben estar integrados en un archivo llamado `patch-`*`xx`* donde *`xx`* marca la secuencia en que los parches serán aplicados — éstos se aplican en *órden alfabético*, así primero será `aa`, segundo `ab` ,etc. Estos archivos deben estar en `PATCHDIR`, desde donde serán automáticamente aplicados. Todos los parches deben ser relativos a `WRKSRC` (generalmente es el directorio en el que se descomprime el port, siendo éste el lugar donde se compila). Para hacer que la solución de problemas y actualizaciones sea más sencilla, deberís evitar tener más de un parche aplicable a un mismo archivo (por ejemplo, `patch-aa` y `patch-ab` modificando ambos al archivo `WRKSRC/foobar.c`).

4.7.2.5. Configurando

Incluye cualquier proceso de personalización adicional en el script `configure` y guardalo en el subdirectorio `scripts`. Como se ha mencionado anteriormente, también puedes hacer esto como un objeto del `Makefile` y/o un script con el nombre `pre-configure` o `post-configure`.

4.7.2.6. Gestionando las entradas de usuario

Si tu port requiere entradas por parte del usuario para la compilación, configuración o instalación, activa la variable `IS_INTERACTIVE` en tu Makefile. Esto permitirá no compilar tu port si el usuario usa la variable BATCH en su entorno (y si el usuario activa la variable INTERACTIVE, entonces *sólo* serán compilados los ports que requieren interacción por parte del usuario).

También es recomendable desactivar el script interactivo si un número razonable de respuestas por defecto son aplicables al port (consultar la variable `PACKAGE_BUILDING`). Esto nos permitirá compilar el package para los CD-ROMs y ftp.

4.7.3. Configurando el Makefile

Configurar el Makefile es bastante sencillo, y de nuevo, te sugerimos que consultes los ejemplos existentes antes de empezar. También hay un Makefile de ejemplo en este handbook, así que, consultalo y, por favor, sigue el órden de las variables y secciones del ejemplo para que tu port sea más fácil de leer para otras personas.

Ahora, considera los siguientes problemas a medida que haces el diseno de tu nuevo Makefile:

4.7.3.1. Los fuentes originales

Reside en `DISTDIR` como un tarball estándar en formato gzip? Si es así puedes pasar al siguiente paso. En caso contrario, deberías considerar la posibilidad de sobreescribir alguna de las variables `EXTRACT_CMD`, `EXTRACT_BEFORE_ARGS`, `EXTRACT_AFTER_ARGS`, `EXTRACT_SUFX`, o `DISTFILES`, dependiendo del formato del archivo de distribución del port. (El caso más común es `EXTRACT_SUFX=.tar.Z`, cuando el tarball está comprimido con compress y no con gzip.)

En el peor de los casos, puedes crearte tu propio objeto `do-extract` para sobreescribir el objeto por defecto.

4.7.3.2. `DISTNAME`

El valor de `DISTNAME` debe ser el nombre base del port. Las reglas por defecto esperan que la lista de archivos de la distribución (`DISTFILES`) se llame `DISTNAMEEXTRACT_SUFX`, el cual, si es un tarball normal, seráa algo como `foozolix-1.0.tar.gz` para usar el valor `DISTNAME=foozolix-1.0`.

Las reglas por defecto también esperan extraer el tarball en un subdirectorio llamado `work/DISTNAME`, por ejemplo `work/foozolix-1.0/`.

Todos estos valores pueden sobreescribirse; simplemente muestran los valores más habituales. Para un port que requiera múltiples archivos de distribución, simplemente usa explícitamente `DISTFILES`. Si solo una parte de los `DISTFILES` son archivos extraibles, entonces usa `EXTRACT_ONLY` para referenciarlos, lo que sobreescribirá la lista `DISTFILES` cuando realice la extracción, y el resto de archivos se dejarán en `DISTDIR` tal y como están para su uso posterior.

4.7.3.3. `PKGNAME`

Si `DISTNAME` no sigue las normas de nombre para un package, deberís asignar a la variable `PKGNAME` un valor más correcto.

4.7.3.4. `CATEGORIES`

Cuando se crea un package, se incluye en el directorio `/usr/ports/packages/All` creandose links desde uno o más subdirectorios de `/usr/ports/packages`. Los nombres de estos subdirectorios están especificados por la variable `CATEGORIES`. Se utiliza para facilitar al usuario la orientación dentro de la colección de ports y packages. Por favor, consulta las categorías existentes y elige las que sean aplicables a tu port.

Esta lista también determina en que lugar del árbol de ports se importa el port. Si especificas más de una categoría, se asume que los archivos del port estarán en el subdirectorio con el nombre en la primera categoría. Consulta la sección categorías para saber como escoger la catergoría adecuada.

Si tu port pertenece realmente a una categorín inexistente, puedes crear una nueva. En este caso, por favor, envía un mail a Lista de Ports de FreeBSD <freebsd-ports@FreeBSD.ORG> para proponer una nueva categoría.

> **Nota:** No hay chequeo de error para los nombres de categorías. `make package` creará un nuevo directorio si tecleas mal el nombre de la categorí, así que se cuidadoso!

4.7.3.5. `MASTER_SITES`

Guarda la parte del directorio del URL -ftp/hhtp que apunta al tarball original en `MASTER_SITES`. No olvidar la barra final (`/`)!

Las macros `make` intentaran usar esta especificación para obtener el archivo de distribución con `FETCH` si no lo pueden encontrar en el sistema.

Se recomienda poner diferentes servidores en la lista, preferiblemente de continentes diferentes. Esto protegerá de posibles problemas de red en grandes áreas, y se está pensando en la posibilidad de anadir soporte para determinar automáticamente el servidor principal más cercano para obtener el archivo desde él.

Si el tarball original forma parte de uno de estos servidores populares: X-contrib, GNU, Perl CPAN, TeX CTAN o Linux Sunsite, se debe hacer referencia a estos servidores usando `MASTER_SITE_XCONTRIB`, `MASTER_SITE_GNU`, `MASTER_SITE_PERL_CPAN`, `MASTER_SITE_TEX_CTAN`, y `MASTER_SITE_SUNSITE`. Simplemente poner `MASTER_SITE_SUBDIR` al path interno del servidor. Aquí hay un ejemplo:

```
MASTER_SITES=         ${MASTER_SITE_XCONTRIB}
MASTER_SITE_SUBDIR=   applications
```

El usuario también puede usar las variables `MASTER_SITE_*` en el archivo `/etc/make.conf` para sobreescribir nuestras selecciones, y usar en su lugar el mirror que prefiera de estos servidores.

4.7.3.6. `PATCHFILES`

Si el port requiere parches adicionales que están disponibles por ftp o http, poner `PATCHFILES` a los nombres de archivos y `PATCH_SITES` a la URL del directorio que los contiene (el formato es el mismo que en `MASTER_SITES`).

Si el parche no es relativo a la raíz del árbol (`WKRSRC`) porque contiene alguna información extra sobre paths, usar la variable `PATCH_DIST_STRIP`. Por ejemplo, si todos los path del parche contienen `foozolix-1.0/` delante del nombre de archivo, usar `PATCH_DIST_STRIP=-p1`.

No preocuparse si los parches están comprimidos, serán descomprimidos automáticamente si el nombre de archivo termina en `.gz` o `.Z`.

Si el parche se distribuye con otros archivos, como documentación, en un tarball gzip, no se puede usar `PATCHFILES`. Si este es el caso, anadir el nombre y situación del tarball a `DISTFILES` y `MASTER_SITES`. A continuación, desde el objeto `pre-patch`, aplicar el parche ejecutando mandato `patch` desde él, el copiando al archivo del parche en el directorio `PATCHDIR` llamandolo `patch-`*`xx`*.

> **Nota:** Tener en cuenta que el tarball será extraido del fuente regular, por lo que no es necesario extraerlo explícitamente si es un tarball comprimido con gzip o compress. Si se hace lo último, tener cuidado de no sobreescribir algo ya existente en ese directorio. No olvidarse de añadir un mandato para borrar el parche copiado en el objeto `pre-clean`

4.7.3.7. `MAINTAINER`

Poner aquí la dirección de mail. Por favor. *:)*

Para una descripción detallada de la responsabilidad de los "maintainers", consultar la sección MAINTAINER en Makefiles.

4.7.3.8. Dependencias

Muchos ports dependen de otros. Hay cinco variables que se pueden usar para asegurar que existen todos los requerimientos necesarios en la máquina del usuario. También hay algunas variables de dependencia pre-soportadas para casos comunes, mas algunos controles de dependencia.

4.7.3.8.1. `LIB_DEPENDS`

Esta variable especifica las librerías compartidas de las que depende el port. Es una lista de registros de *`lib`*:*`dir`*[*`:target`*] donde *`lib`* es el nombre de la librerí compartida, y *`dir`* es el directorio en el cual encontrarla en caso de no estar disponible y *`target`* es el objeto a llamar en ese directorio. Por ejemplo,

```
LIB_DEPENDS=jpeg.9:${PORTSDIR}/graphics/jpeg:install
```

comprobará la existencia de la librería jpeg versión 9, descendiendo al subdirectorio `graphics/jpeg` de la colección de ports para compilarlo e instalarlo en caso de que no sea encontrado en el sistema. El objeto *`target`* puede omitirse si es igual a `DEPENDS_TARGET` (el cual, por defecto, tiene el valor `install`).

> **Nota:** La parte *`lib`* es un argumento dado a `ldconfig -r | grep -wF`. Es posible que no existan expresiones regulares en esta variable.

La dependencia se comprueba dos veces, una durante la ejecución de `extract` y otra durante la ejecución de `install`. También, se incluye el nombre de la dependencia en el package, de manera que `pkg_add` la instale automáticamente si no está en el sistema del usuario.

4.7.3.8.2. `RUN_DEPENDS`

Esta variable especifica ejecutables o archivos de los que depende la compilació y/o ejecución de este port. Es una lista de tuples *`path`*:*`dir`*[*`:target`*] donde *`path`* es el nombre del ejecutable o archivo, *`dir`* es el directorio en el que encontrarlo en caso de no estar disponible y *`target`* es el objeto a llamar en ese directorio. Si *`path`* empieza con una barra (`/`), se trata como un archivo comprobando su existencia con `test -e`; en caso contrario, se asume que es un ejecutable, y se usa `which -s` para determinar si el programa existe en el path del usuario.

Por ejemplo,

```
RUN_DEPENDS=   ${PREFIX}/etc/innd:${PORTSDIR}/news/inn \
               wish8.0:${PORTSDIR}/x11-toolkits/tk80
```

comprobará si existe el directorio o archivo `/usr/local/etc/innd`, compilando e instalandolo desde el directorio `news/inn` del árbol de ports en caso de no existir. También se comprueba la existencia del ejecutable `wish8.0` en el path, descendiendo al subdirectorio `x11-toolkits/tk80` de los ports para compilarlo e instalarlo si no existe.

> **Nota:** En este caso, `innd` es un ejecutable; si un ejecutable está situado en un lugar diferente al esperado (fuera del path), es necesario incluir el path completo.

La dependencia se comprueba desde el objeto `install`. El nombre de la dependencia también es incluido en el package para que `pkg_add` lo instale automáticamente si no existe en el sistema del usuario. La parte *`target`* puede omitirse si es el mismo `DEPENDS_TARGET`.

4.7.3.8.3. `BUILD_DEPENDS`

Esta variable especifica los ejecutables o archivos que este port necesita para compilar. Igual que `RUN_DEPENDS`, es una lista de tuples *`path`*:*`dir`*[*`:target`*]. Por ejemplo,

```
BUILD_DEPENDS=
       unzip:${PORTSDIR}/archivers/unzip
```

comprobará la existencia de un ejecutable llamado `unzip`, descendiendo al subdirectorio `archivers/unzip` en caso de no existir, para compilarlo e instalarlo.

> **Nota:** Compilación en este caso, se refiere a todo el proceso; desde la extracción hasta la compilación final. La dependencia se comprueba en el objeto `extract`. La parte *`target`* puede omitirse si la misma que `DEPENDS_TARGET`

4.7.3.8.4. `FETCH_DEPENDS`

Esta variable especifica ejecutables o archivos que se requieren para obtener este port. Como los dos anteriores, es una lista de tuples *`path`*:*`dir`*[*`:target`*]. Por ejemplo,

```
FETCH_DEPENDS=
       ncftp2:${PORTSDIR}/net/ncftp2
```

comprobará la existencia de un ejecutable llamado `ncftp2`, descendiendo al subdirectorio `net/ncftp2` de los ports para compilarlo e instalarlo en caso de no existir.

La dependencia es comprobada en el objeto `fetch`. La parte *`target`* puede omitirse si es la misma que `DEPENDS_TARGET`.

4.7.3.8.5. `DEPENDS`

Si hay una dependencia que no se puede incluir en las cuatro categorías anteriores, o el port necesita tener extraidos los fuentes de otro port para poder ser instalado, usar esta variable. Esta es una lista compuesta de *`dir`*[*`:target`*], ya que no no hay nada que comprobar, al contrario que las cuatro anteriores. La parte *`target`* puede omitirse si es la misma que `DEPENDS_TARGET`.

4.7.3.8.6. Variables de dependencia comunes

Definir `USE_XLIB=yes` si el port requiere la instalación del sistema X Window (implícito en `USE_IMAKE`). Definir `USE_GMAKE=yes` si el port requiere el `make` de GNU en lugar del `make` de BSD. Definir `USE_AUTOCONF=yes` si el port requiere la ejecución del autoconf de GNU. Definir `USE_QT=yes` si el port usa el último kit qt. Usar

`USE_PERL5=yes` si el port requiere la versión 5 del lenguaje perl. (Este último es especialmente importante ya que unas versiones de FreeBSD contienen perl5 como parte del sistema y otras no.)

4.7.3.8.7. Notas sobre dependencias

Cómo se ha mencionado anteriormente, el objeto por defecto a llamar cuando se require una dependencia es `DEPENDS_TARGET`. Por defecto es `install`. Esta es una variable de usuario; nunca se define en los `Makefile` de los ports. Si un port necesita gestionar las dependencias de manera especial, usar la parte `:target` de las variables `*_DEPENDS` en lugar de redefinir `DEPENDS_TARGET`.

Cuando se ejecuta `make clean`, sus dependencias también son "limpiadas". Si se quiere evitar esto, hay que definir la variable `NOCLEANDEPENDS` en el entorno.

Para depender incondicionalmente de un port, es imprescindible usar la cadena de texto `nonexistent` como primer campo de `BUILD_DEPENDS` o `RUN_DEPENDS`. Usar esto sólo cuando es necesario disponer del código fuente de otro port. Se puede ahorrar tiempo de compilación especificando el objeto. Por ejemplo:

```
BUILD_DEPENDS=   /nonexistent:${PORTSDIR}/graphics/jpeg:extract
```

siempre descenderá al port JPEG y lo extraerá.

No usar `DEPENDS` a no ser que no exista otra manera de obtener los resultados deseados. Hará que el otro port siempre sea compilado (e instalado, por defecto), y la dependencia sea anadia al package. Si esto es lo que realmente se necesita, es recomendable usar `BUILD_DEPENDS` y `RUN_DEPENDS`.

4.7.3.9. Mecanismos de creación

Si el package usa GNU `make`, definir `USE_GMAKE=yes`. Si el package usa `configure`, definir `HAS_CONFIGURE=yes`. Si el package usa GNU `configure`, definir `GNU_CONFIGURE=yes` (esto implica `HAS_CONFIGURE`). Si se quieren pasar argumentos extra a `configure` (el argumento por defecto es `-prefix=${PREFIX}` para GNU `configure` y vacío para no GNU `configure`), definir los argumentos extra en `CONFIGURE_ARGS`. Si el package usa GNU `autoconf`, definir `USE_AUTOCONF=yes`. Esto implica `GNU_CONFIGURE`, y causará la ejecución de `autoconf` antes que `configure`.

Si el package es una aplicación X que crea archivos `Makefile`s desde `Imakefile`s usando `imake`, definir `USE_IMAKE=yes`. Esto hará que durante el proceso de configuración se ejecute el mandato `xmkmf -a`. Si el argumento `-a` es problemático para el port, definir `XMKMF=xmkmf`. Si el port usa el mandato `imake` pero no entiende el objeto `install.man`, definir `NO_INSTALL_MANPAGES=yes`.

Si el código fuente incluido en el `Makefile` del port tiene algo más que el objeto `all` como objeto principal de creación, definir `ALL_TARGET` correctamente. Hacer lo mismo con `install` y `INSTALL_TARGET`.

4.7.4. Consideraciones especiales

Hay más cosas a tener en cuenta a la hora de crear un port. Esta sección explica las más comunes.

4.7.4.1. `ldconfig`

Si el port instala una librería compartida, anadir un objeto `post-install` al `Makefile` para que ejecute `${LDCONFIG} -m` en el directorio donde se ha instalado la nueva librería (normalmente `PREFIX/lib`) para registrarla en el caché de librerías compartido.

También, anadir un par `@exec /sbin/ldconfig -m` y `@unexec /sbin/ldconfig -R` al archivo `pkg/PLIST` para que el usuario que ha instalado el port pueda usar la librería compartida inmediatamente. Estas líneas deben estar inmediatamente después de la línea de la propia librería compartida, como en:

```
lib/libtvl80.so.1
@exec /sbin/ldconfig -m %D/lib
@unexec /sbin/ldconfig -R
```

Nunca, nunca, *nunca* anadir una línea que ponga `ldconfig` sin argumentos en el `Makefile` o `pkg/PLIST`. Esto hará que el caché de librerías compartidas se resetee sólo con los contenidos de `/usr/lib`, probablemente, dejando el sistema inestable ("Ayuda, xinit no ha vuelto a funcionar desde que instalé este port"). Cualquiera que haga esto, será troceado en 65.536 partes con un cuchillo poco afilado, permaneciendo por toda la eternidad en lo más profundo del infierno (y no necesariamente en este órden)

4.7.5. Soporte ELF

Desde la transición de FreeBSD a formato ELF después de la version 3.0-RELEASE, fue necesario convertir muchos ports para que generasen librerías compartidas en formato ELF. Para complicar más el trabajo, los sistemas 3.0 pueden ejecutar ELF y a.out, y se quiere mantener, etraoficialmente, tanto como sea posible, el soporte para versiones 2.2. A continuación se explica como convertir ports que sólo soportan a.out para que soporten compilaciones a.out y ELF.

Una parte de esta lista sólo es aplicable durante la conversión, pero se mantendrá durante un tiempo como referencia en caso de querer actualizar algún port antiguo.

4.7.5.1. Mover las librerías a.out

Las librerías a.out deben moverse de `/usr/local/lib` y similares a un subdirectorio `aout`. (Si no se mueven, los ports ELF sobreescribirán las librerías a.out). El objeto `move-aout-libs` del archivo `src/Makefile` en 3.0-CURRENT (llamado desde `aout-to-elf`) hará este paso en nuestro lugar. Sólo moverá las librerías a.out en los directorios estándar.

4.7.5.2. Formato

La colección de ports compilará los packages en el mismo formato en el que esté la máquina. Esto significa a.out para 2.2 y a.out o ELF para 3.0 dependiendo de lo que devuelva la variable `‘objformat‘`. De la misma manera, una vez se mueven las librerías a un subdirectorio, ya no será posible compilar librerías a.out.

> **Nota:** Si un port sólo funciona para a.out, definir `BROKEN_ELF`. Estos ports serán pasados por alto durante la compilación en sistemas ELF.

4.7.5.3. `PORTOBJFORMAT`

`bsd.port.mk` definirá `PORTOBJFORMAT` a `aout` o `elf` y lo exportará en las variables de entorno CONFIGURE_ENV, SCRIPTS_ENV y MAKE_ENV. (Siempre será `aout` en 2.2-STABLE). También se pasa a `PLIST_SUB` como `PORTOBJFORMAT=${PORTOBJFORMAT}`. (Consultar los comentarios sobre `ldconfig` de las líneas anteriores).

La variable se defina usando esta línea en `bsd.port.mk`:

```
PORTOBJFORMAT!= test -x /usr/bin/objformat && /usr/bin/objformat || echo aout
```

Los procesos de compilación de los ports deberían usar esta variable para decidir que hacer. De todas maneras, si el script `configure` del port detecta automáticamente un sistema ELF, no es necesario hacer referencia a `PORTOBJFORMAT`.

4.8. porting-pkgname (Incompleto)

Incompleto

4.9. contrib-general (Incompleto)

Incompleto

4.10. porting-cleaning (Incompleto)

porting-cleaning

4.11. porting-samplem

porting-samplem

4.12. porting-dads

porting-dads

4.13. porting-categories

porting-categories

II. Administración del sistema

Capítulo 5. Configurando el Kernel de FreeBSD

Configuracion Kernel

5.1. El archivo de configuración

a

Capítulo 6. Seguridad

Seguridad

Capítulo 7. Impresion

Impresion

Capítulo 8. Discos

Escrito por David O'Brien <obrien@FreeBSD.org> 26 de Abril de 1998

Supongamos que queremos anadir un nuevo disco SCSI a una máquina que en la actualidad sólo tiene un único disco. En primer lugar, apague el ordenador e instale el disco siguiendo las instrucciones del fabricante del ordenador, de la controladora y del propio disco. A causa de la diversidad de procedimientos para hacer esto, los detalles exceden el ámbito de este documento.

Entre en el sistema como el usuario `root`. Una vez que haya instalado el disco, inspeccione `/var/run/dmesg.boot` para asegurarse de que el nuevo disco ha sido encontrado. Continuando con nuestro ejemplo, el disco recientemente anadido será `da1` y lo montaremos en `/1` (si está usted anadiendo un disco IDE, sustituya `da` por `wd`).

Como FreeBSD funciona en ordenadores compatibles con el IBM-PC, debe tener en cuenta las particiones de la BIOS del PC. Estas son diferentes de las particiones tradicionales de tipo BSD. Un disco de PC puede tener hasta cuatro entradas para particiones de tipo BIOS. Si el disco va a ser realmente dedicado a FreeBSD, puede usted emplear el modo dedicado (*dedicated*). En otro caso, FreeBSD tendrá que alojarse en una de las particiones de la BIOS. FreeBSD llama a las particiones de la BIOS *slices*, para no confundirlas con las particiones tradicionales de tipo BSD. También puede usted usar slices en un disco que esté dedicado a FreeBSD, pero en un ordenador que además tenga otro sistema operativo instalado. Esto es así para no confundir a la utilidad `fdisk` del otro sistema operativo.

En el caso de utilizar slices, el disco será anadido como `/dev/da1s1e`. Esto se lee del siguiente modo: disco SCSI, unidad número 1 (segundo disco SCSI), slice 1 (partición 1 de la BIOS) y partición `e` de tipo BSD. En el caso de utilizar el modo dedicado, el disco será anadido, simplemente, como `/dev/da1e`.

8.1. Utilizando Sysinstall

Puede utilizar `/stand/sysinstall` para crear y etiquetar particiones en un disco nuevo usando sus intuitivos menús. Bien entre en el sistema como el usuario `root`, o bien utilice el comando `su`. Ejecute `/stand/sysinstall` y seleccione el menú `Configure`. Dentro del menú `FreeBSD Configuration Menu`, baje hasta seleccionar la opción `Partition`. A continuación, se le debería mostrar una lista de los discos duros instalados en su sistema. Si no aparece `da1`, deberá usted comprobar la instalación física del disco y la salida del comando `dmesg` almacenada en el archivo `/var/run/dmesg.boot`.

Seleccione `da1` para pasar al editor de particiones `FDISK Partition Editor`. Pulse `A` para usar el disco entero para FreeBSD. Cuando se le pregunte si quiere mantener el disco compatible con la instalación de algún sistema operativo en el futuro ("remain cooperative with any future possible operating systems") conteste afirmativamente (`YES`). Escriba los cambios en el disco pulsando `W`. Ahora salga del editor FDISK pulsando `q`. A continuación se le preguntará acerca del sector de inicio del disco (MBR, Master Boot Record). Como está usted anadiendo un disco a un sistema que ya se encuentra en funcionamiento, seleccione la opción `None` (esta opción deja el MBR intacto).

A continuación llegará al `Disk Label Editor`. Aquí es donde creará las particiones de estilo tradicional BSD. Un disco puede tener hasta ocho particiones, designadas a-h. Algunas de las etiquetas de las particiones tienen usos especiales. La etiqueta `a` se utiliza para la partición raíz (`/`, root partition). Por tanto, sólo su disco de sistema (es decir, aquel desde el cual arranca el sistema) debería tener una partición `a`. La etiqueta `b` se utiliza para particiones de swap (memoria virtual) y es posible tener varios discos con particiones de swap. La etiqueta `c` representa a todo el disco utilizado en modo dedicado, o a todo el slice de FreeBSD si es que utilizamos el modo slice. Las restantes etiquetas son para uso general.

El editor de etiquetas (de particiones) de Sysinstall utiliza la etiqueta `e` para particiones que no son root ni swap. Dentro del editor de etiquetas de particiones, cree un único sistema de archivos pulsando `C`. Cuando se le pregunte si será un sistema de archivos (FS, file system) o swap, elija `FS` e indique un punto donde montarlo (por ejemplo, `/mnt`). Cuando anada un disco una vez instalado FreeBSD, Sysinstall no creará la entrada correspondiente en `/etc/fstab`, por lo que no resulta importante el punto que indique para montar el disco.

En este momento, ya puede escribir en el disco la información de las particiones y sus etiquetas y crear un sistema de archivos. Para ello pulse `W`. Ignore cualquier error de Sysinstall que le informe de que no pudo montar la nueva partición. Ya puede salir del editor de etiquetas de particiones y de Sysinstall.

El último paso consiste en editar `/etc/fstab` para anadir una entrada para el nuevo disco.

8.2. Usando la línea de comandos

8.2.1. * Utilizando Slices

8.2.2. Modo Dedicado

Si no va a compartir el nuevo disco con otro sistema operativo, puede usar el modo dedicado (`dedicated`). Recuerde que este modo puede confundir a los sistemas operativos de Microsoft; no obstante, estos no causarán ningún dano. En cambio, el OS/2 de IBM se “apropiará” de cualquier partición que encuentre y que no reconozca.

```
# dd if=/dev/zero of=/dev/rda1 bs=1k count=1
# disklabel -Brw da1 auto
# disklabel -e da1 # creamos la partición ‘e’
# newfs -d0 /dev/rda1e
# mkdir -p /1
# vi /etc/fstab # añadimos una entrada para /dev/da1e
# mount /1
```

Un método alternativo resulta:

```
# dd if=/dev/zero of=/dev/rda1 count=2
# disklabel /dev/rda1 | disklabel -BrR da1 /dev/stdin
# newfs /dev/rda1e
# mkdir -p /1
# vi /etc/fstab # añadimos una entrada para  /dev/da1e
# mount /1
```

8.3. * Discos no tradicionales

8.3.1. * Discos Zip

8.3.2. * Discos Jaz

8.3.3. * Discos Sequest

Capítulo 9. Backups

Las cuestiones de compatibilidad de hardware se encuentran entre las más problemáticas hoy en día en la industria de los componentes de ordenador y FreeBSD no es, de ningún modo, inmune a estos problemas. A este respecto, la ventaja de FreeBSD consistente en su capacidad para funcionar sobre componentes normales de PC supone una carga a la hora de soportar la sorprendente variedad de componentes que hay en el mercado. Si bien sería imposible suministrar una lista exhaustiva de componentes que FreeBSD soporta, esta sección sirve de catálogo de los controladores de dispositivo incluidos con FreeBSD y de los componentes que cada controlador soporta. Donde resulta posible y adecuado, se incluye notas acerca de productos específicos. Puede usted consultar la sección de configuración del núcleo, en este mismo manual, donde encontrará una lista de los dispositivos soportados.

Como FreeBSD es un proyecto realizado por voluntarios y carece de un departamento de pruebas financiado, dependemos de usted, el usuario, para gran parte de la información contenida en este catálogo de dispositivos. Si tiene usted experiencia directa de componentes que funcionen o que no funcionen con FreeBSD, por favor notifíquenoslo mediante correo electrónico a la lista Lista del Proyecto de Documentación de FreeBSD <freebsd-doc@FreeBSD.ORG>. Las preguntas acerca de los dispositivos soportados pueden dirigirse a la lista Lista de preguntas generales de FreeBSD <freebsd-questions@FreeBSD.ORG> (consulte Listas de Correo para más información). Cuando envíe información o realice una pregunta, por favor recuerde especificar exactamente qué versión de FreeBSD está utilizando, así como incluir tantos detalles de los componentes de su ordenador como sea posible.

9.1. Soportes de cinta

Los principales soportes de cinta son: 4 mm, 8 mm, QIC, mini-cartuchos y DLT.

9.1.1. 4 mm (DDS: Digital Data Storage, Almacenamiento digital de datos)

Las cintas de 4 mm están sustituyendo a las QIC como el soporte de copias de seguridad más popular. Esta tendencia se aceleró considerablemente cuando Conner compró Archive, un destacado fabricante de unidades QIC, y entonces detuvo la producción de unidades QIC. Las unidades de 4 mm son pequenas y silenciosas, pero no tienen la reputación de fiabilidad de las unidades de 8 mm. Los cartuchos son menos caros y más pequenos (76 x 51 x 12 mm, 3 x 2 x 0.5 pulgadas) que los cartuchos de 8 mm. Las unidades de 4 mm, como las de 8 mm, presentan una vida comparativamente corta del cabezal por el mismo motivo, ambas emplean un barrido helicoidal.

La tasa de transferencia de datos en estas unidades comienza en torno a 150 kB/s, alcanzando picos de 500 kB/s. La capacidad de almacenamiento comienza en 1.3 GB y llega hasta 2.0 GB. La compresión mediante hardware, disponible en la mayoría de estas unidades, duplica aproximadamente la capacidad. Las unidades de cinta múltiples

pueden constar de 6 unidades en un mismo chasis, con funciones de cambio automático de cintas. La capacidad de las librerías (de cintas) alcanzan 240 GB.

Las unidades de 4 mm, como las de 8 mm, emplean un barrido helicoidal. Todas las ventajas e inconvenientes del barrido helicoidal son aplicables tanto a las unidades de 4 mm, como a las de 8 mm.

Las cintas deberían ser retiradas de uso tras 2000 lecturas o 100 copias de seguridad completas.

9.1.2. 8 mm (Exabyte)

Las unidades de 8 mm son las más comunes para controladoras SCSI; son la mejor elección para intercambiar cintas. En casi cualquier sistema se puede encontrar una unidad Exabyte de 8 mm con 2 GB de capacidad. Las unidades de 8 mm son fiables, cómodas y silenciosas. Los cartuchos son baratos y pequenos (122 x 84 x 15 mm; 4.8 x 3.3 x 0.6 pulgadas). Un inconveniente de las cintas de 8 mm es la vida relativamente corta de la cinta y de los cabezales, debido a la alta tasa de movimiento relativo entre ambos.

La tasa de transferencia de datos oscila, aproximadamente, entre 250 kB/s y 500 kB/s. La capacidad de almacenamiento comienza en 300 MB y alcanza 7 GB. La compresión mediante hardware, disponible en la mayoría de estas unidades, aproximadamente dobla la capacidad. Esta unidades están disponibles como unidades independientes y como unidades de cinta múltiples, con 6 unidades y 120 cintas en un mismo chasis. Las cintas son cambiadas automáticamente por la unidad. La capacidad de las librerías alcanza más de 840 GB.

La información se registra en la cinta empleando para ello un barrido helicoidal, los cabezales se sitúan formando un cierto ángulo con la cinta (aproximadamente 6 grados). La cinta rodea 270 grados al soporte que sostiene los cabezales. El soporte gira mientras la cinta se desliza sobre ella. El resultado es una gran densidad de datos y las pistas dispuestas muy cerca entre sí y a través de la cinta, formando un ángulo, desde un extremo hasta el otro.

9.1.3. QIC

Las cintas y las unidades QIC-150 son, quizás, las más comunes. Las unidades QIC son las menos caras de entre las unidades de backup "serias". El inconveniente es el coste de las cintas. Las cintas QIC son caras comparadas con las de 8mm o las de 4mm, hasta 5 veces más caras por GB. Pero si sus necesidades se pueden satisfacer con media docena de cintas, QIC puede ser una elección acertada. Las unidades QIC son las unidades de cinta *más* comunes. En todos los sistemas hay una unidad QIC de alguna densidad. Ahi reside el problema, QIC presenta un gran número de densidades en cintas físicamente similares (en ocasiones, idénticas). Las unidades QIC no son silenciosas. Estas unidades resultan audibles cuando se sitúan antes de comenzar a escribir y claramente audibles cuando leen, escriben o se sitúan (seek). Las dimensiones de las cintas QIC son: 15.2 x 10.2 x 1.7 mm (6 x 4 x 0.7 pulgadas). Los mini-cartuchos, que también utilizan cinta de 1/4 de pulgada de anchura, se comentan en otro apartado. No están disponibles ni cambiadores, ni unidades múltiples de cintas.

La tasa de transferencia de datos oscila entre, aproximadamente, 150k B/s y 500kB/s. La capacidad de almacenamiento oscila ente 40 MB y 15 GB. La compresión mediante hardware está disponible en muchas de las nuevas unidades QIC. Las unidades QIC se instalan cada vez menos; están siendo desplazadas por las unidades DAT.

La información se almacena en pistas. Las pistas están dispuestas longiudinalmente en la cinta, de un extremo al otro. El número de pistas, y por tanto la anchura de las mismas, varía en función de la capacidad de la cinta. Si no todas, la mayoría de las unidades nuevas presentan compatibilidad con las anteriores al menos para la lectura (frecuentemente, también para la escritura). QIC tiene una buena reputación en relación con la seguridad de los datos (el mecanismo es más simple y más robusto que el de las unidades con barrido helicoidal).

Las cintas deberín ser retiradas de uso tras 5000 copias de seguridad.

9.1.4. * Mini-Cartuchos

9.1.5. DLT

Las unidades DLT presentan la tasa de transferencia de datos más elevada de todos los tipos de unidades comentados. La cinta de 12.5 mm (1/2 pulgada) está contenida en un cartucho de una única bobina (100 x 100 x 25 mm; 4 x 4 x 1 pulgadas). El cartucho tiene una puerta que se abate a lo largo de un costado del mismo. El mecanismo de la unidad abre esta puerta para extraer la guía de la cinta. La guía de la cinta tiene un agujero oval que es usado por la unidad para "enganchar"la cinta. La bobina en la que se va a enrollar la cinta se localiza en el interior de la unidad. Los restantes cartuchos de cinta mencionados (con la única excepción de las cintas de 9 pistas) tienen situadas en el interior del propio cartucho ambas bobinas, tanto aquella en la que está inicialmente enrollada la cinta, como la bobina en la que se va a enrollar.

La tasa de transferencia de datos es aproximadamente 1.5 MB/s, tres veces la de las unidades de 4 mm, de 8 mm y QIC. La capacidad oscila entre 10 GB y 10 GB para una unidad sencilla. Las unidades se encuentran disponibles como unidades sencillas con cambiador de cintas y también como unidades múltiples con cambiador de cintas, conteniendo desde 5 hasta 900 cintas, desde 1 a 20 unidades y siendo capaces de almacenar entre 50 GB y 9 TB.

La información se graba en la cinta en pistas paralelas a la dirección de desplazamiento (como en las cintas QIC). Se escriben dos pistas a la vez. La vida útil de los cabezales de lectura/escritura es relativamente larga; una vez la cinta se detiene, no hay movimiento relativo entre los cabezales y la cinta.

9.1.6. Usando una cinta nueva por primera vez

La primera vez que intente leer o escribir en una cinta nueva y completamente virgen, la operación fracasará. Los mensajes de la consola deberían ser similares a los siguientes:

```
sa0(ncr1:4:0): NOT READY asc:4,1
sa0(ncr1:4:0):  Logical unit is in process of becoming ready
```

La cinta no contiene un bloque de identificación (bloque número 0). Desde la adopción del standard QIC-525, todas las unidades QIC escriben en la cinta un bloque de identificación. Caben dos soluciones:

`mt fsf 1` hace que la unidad escriba un bloque de identificación en la cinta.

Use el botón del panel frontal de la unidad para expulsar la cinta.

Reinserte la cinta y transfiera mediante dump(8) la información a la cinta.

dump(8) mostrará el mensaje `DUMP: End of tape detected` y en la consola aparecerá: `HARDWARE FAILURE info:280 asc:80,96`

rebobine la cinta utilizando: `mt rewind`

Las siguientes operaciones sobre la cinta tendrán éxito.

9.2. Programas para hacer copias de seguridad

Los tres programas principales son dump(8), tar(1), y cpio(1).

9.2.1. Dump y Restore

dump(8) y restore(8) son los programas tradicionales de Unix para realizar copias de seguridad. Operan sobre la unidad como si ésta fuese un conjunto de bloques de disco, a un nivel inferior a las abstracciones de archivos, enlaces y directorios, creadas por los sistemas de archivos. dump(8) hace copias de seguridad de dispositivos, sistemas de archivos enteros, no de partes de un sistema de archivos ni de árboles de directorios que se extienden por más de un sistema de archivos utilizando enlaces simbólicos, o bien montando un sistema bajo el otro. dump(8) no escribe en la cinta archivos ni directorios, sino que más bien escribe bloques de datos que son las piezas con las que se construyen los archivos y directorios. dump(8) presenta peculiaridades que provienen de su origen en la Versión 6 del Unix de ATT (en torno a 1975). Los parámetros por defecto resultan adecuados para las cintas de 9 pistas (6250 bpi), pero no para los soportes de alta densidad disponibles hoy en día (hasta 62,182 ftpi). Estos valores por defecto deben ser anulados mediante la línea de comandos para utilizar la capacidad de las unidades de cinta actuales.

rdump(8) y rrestore(8) hacen copias de seguridad a través de la red, en una unidad conectada a otro ordenador. Ambos programas se basan en rcmd(3) y ruserok(3) para acceder a la unidad de cinta remota. Por tanto, el usuario que realiza la copia de seguridad debe tener acceso al ordenador remoto mediante `rhosts`. Los argumentos suministrados a rdump(8) y a rrestore(8) deben ser adecuados para el ordenador remoto. (Por ejemplo, cuando esté volcando datos mediante `rdump` desde un ordenador con FreeBSD a una unidad Exabyte conectada a una estación

Sun, de nombre `komodo`, emplee: `/sbin/rdump 0dsbfu 54000 13000 126 komodo:/dev/nrsa8 /dev/rda0a 2>&1`) Atención: permitir comandos `rhosts` afecta a la seguridad. Evalúe su sitación ciudadosamente.

9.2.2. Tar

tar(1) también se remonta a la Versión 6 del Unix de ATT (hacia 1975). tar(1) trabaja con el sistema de archivos; tar(1) escribe archivos y directorios en la cinta. tar(1) no soporta todas las opciones disponibles para cpio(1), pero tar(1) no necesita la inusual tubería que cpio(1) emplea.

La mayor parte de las versiones de tar(1) no soportan la realización de copias de seguridad a través de la red. La versión GNU de tar(1), la que se usa en FreeBSD, soporta dispositivos remotos empleando la misma sintaxis que rdump(8). Para hacer una copia de seguridad mediante tar(1) a una unidad Exabyte conectada a una estación Sun de nombre `komodo`, se utilizaría: `/usr/bin/tar cf komodo:/dev/nrsa8 . 2>&1`. Para las versiones que no soportan dispositivos remotos, se puede utilizar una tubería y rsh(1) para enviar los datos a la unidad remota.

9.2.3. Cpio

cpio(1) es el programa original de Unix para intercambiar archivos mediante soportes magnéticos. cpio(1) tiene, entre muchas otras, opciones para realizar intercambio (swapping) de bytes, escribir en diferentes formatos de archivo y enviar mediante una tubería los datos a otros programas. Esta última prestación hace de cpio(1) una excelente elección para soportes de instalación. cpio(1) no sabe cómo recorrer el árbol de directorios, por lo que se le debe suministrar una lista de archivos a través de `stdin`.

cpio(1) no soporta la realización de copias de seguridad a través de la red. Puede utilizar una tubería junto con rsh(1) para enviar los datos a una unidad de cinta remota.

9.2.4. Pax

pax(1) es la respuesta de IEEE/POSIX a tar(1) y a cpio(1). A lo largo de los anos, las distintas versiones de tar(1) y de cpio(1) se han vuelto ligeramente incompatibles. Por ello, más que luchar para estandarizarlas completamente, POSIX creó un nuevo programa para realizar copias de seguridad. pax(1) lee y escribe en varios de los diversos formatos de cpio(1) y tar(1), además de nuevos formatos propios. Su juego de comandos se parece más al de cpio(1) que al de tar(1).

9.2.5. Amanda

Amanda (../ports/misc.html#amanda-2.4.0) (Advanced Maryland Network Disk Archiver) constituye, más que un único programa, un sistema con estructura cliente/servidor. Un servidor de Amanda realizará, en una única unidad de

cinta, copias de seguridad de un número indeterminado de ordenadores que tengan clientes de Amanda y comunicación mediante red con el servidor de Amanda. Un problema habitual en aquellos lugares en los que hay muchos discos grandes consiste en que el tiempo necesario para realizar copias de seguridad directamente sobre la cinta excede el tiempo disponible para la tarea. Amanda soluciona este problema. Amanda puede utilizar un "disco contenedor"para realizar copias de seguridad de varios sistemas de archivos a la vez. Amanda crea "juegos de archivos": un grupo de cintas utilizadas por un determinado periodo de tiempo para hacer copias de seguridad completas (full backups) de todos los sistemas de archivos indicados en el archivo de configuración de Amanda. El "juego de archivos"también contiene copias de seguridad incrementales (o diferenciales) de los sistemas de archivos, realizadas diariamente (normalmente por la noche). Restaurar un sistema de archivos danado requiere la copia de seguridad completa más reciente y las copias de seguridad incrementales.

El archivo de configuración permite un control preciso de las copias de seguridad y del tráfico que Amanda genera en la red. Amanda usará cualquiera de los programas mencionados arriba para escribir los datos en la cinta. Amanda se encuentra disponible tanto en forma de port, como en forma de paquete (package). Amanda no se instala por defecto.

9.2.6. No haga nada

“No haga nada” no es un programa de ordenador, pero es la estrategia de copias de seguridad más ampliamente usada. No presenta costes iniciales. No hay que seguir ninguna planificación. Simplemente diga no. Si algo le sucediese a sus datos, sonría y resígnese!

Si su tiempo y sus datos no tienen valor alguno, entonces “No haga nada” es el procedimiento de realización de copias de seguridad más adecuado para usted. Pero sepa que, como Unix es una herramienta útil, puede que en seis meses tenga usted una colección de archivos que le resulte valiosa.

“No haga nada” es el método de realización de copias de seguridad apropiado para `/usr/obj` y otros árboles de directorios que pueden ser exactamente reproducidos por su odenador. Un ejemplo de esto lo constituyen los archivos que contienen las páginas de este manual, que han sido generados a partir de archivos SGML. Hacer copias de seguridad de estos archivos HTML resulta innecesario. De los archivos fuente en formato SGML se realiza regularmente copias de seguridad.

9.2.7. ¿Qué programa es el mejor para realizar copias de seguridad?

dump(8) *Punto.* Elizabeth D. Zwicky probó concienzudamente todos los programas para realizar copias de seguridad aquí comentados. La mejor elección para preservar todos los datos y las peculiaridades de los sistemas de archivos de Unix es dump(8). Elizabeth creó sistemas de archivos que presentaban una gran variedad de condiciones inusuales (y algunas no tan inusuales) y probó cada programa haciendo copias de seguridad de esos sistemas de archivos y restaurándolas. Estas condiciones inusuales incluían: archivos con huecos, archivos con huecos y con un bloque de caracteres NULL, archivos con caracteres extranos en el nombre, archivos sobre los que no se podía leer ni escribir, dispositivos, archivos que cambiaban de tamano durante la realización de la copia de seguridad, archivos que eran

creados/borrados durante la realización de la copia de seguridad, etc. Elisabeth presentó los resultados de su estudio en LISA V, en octubre de 1991. Véase Torture-testing Backup and Archive Programs (http://reality.sgi.com/zwicky_neu/testdump.doc.html).

9.2.8. Procedimiento de restauración de emergencia

9.2.8.1. Antes del desastre

Hay cuatro pasos que necesita realizar para prepararse para cualquier desastre que pudiera ocurrir.

En primer lugar, imprima la tabla de particiones BSD (disklabel) de cada uno de sus discos (por ejemplo, `disklabel da0 | lpr`), su tabla de sistemas de archivos (`/etc/fstab`) y todos los mensajes que aparecen al iniciar el sistema operativo. Haga dos copias de cada uno.

En segundo lugar, compruebe que los discos de inicio y de reparación (`boot.flp` y `fixit.flp`) contienen controladores para todos sus dispositivos. El modo más sencillo de comprobarlo es reiniciar el ordenador con el disco de arranque en la disquetera y comprobar los mensajes que aparecen al iniciarse el sistema operativo. Si todos sus dispositivos aparecen mencionados y están operativos, pase al punto tercero.

De no ser así, deberá crear a medida dos discos de inicio que contengan un núcleo que permita montar todos sus discos y acceder a su unidad de cinta. Estos discos deben contener: fdisk(8), disklabel(8), newfs(8), mount(8), y cualquier otro programa que utilice para realizar copias de seguridad. Estos programas deben estar enlazados estáticamente. Si utiliza dump(8), el disquete deberá contener también restore(8).

En tercer lugar, realice copias de seguridad regularmente. Cualquier cambio que haga tras su última copia de seguridad puede resultar perdido sin posibilidad de recuperación. Proteja contra escritura las cintas que contengan las copias de seguridad.

En cuarto lugar, pruebe los discos (bien `boot.flp` y `fixit.flp` o bien los dos discos que creó a medida en el segundo paso) y las cintas que contienen las copias de seguridad. Tome notas del proceso. Almacene esas notas junto con los discos de arranque, los datos que imprimió y las cintas que contienen las copias de seguridad. Se encontrará tan inquieto cuando proceda a la restauración que las notas pueden impedir que destruya las cintas que contienen las copias de seguridad (¿Cómo? En lugar de `tar xvf /dev/rsa0`, usted podría accidentalmente teclear `tar cvf /dev/rsa0` y sobreescribir las cintas que contienen las copias de seguridad).

Como medida adicional de seguridad, haga discos de arranque y copias por duplicado cada vez. Guarde uno de cada en un lugar lejano. Un lugar lejano NO es el sótano del mismo edificio. Varias companías aprendieron esta lección por las malas en el Centro Mundial del Comercio (World Trade Center). Un lugar lejano debería estar físicamente separado de sus ordenadores y discos duros por una distancia considerable.

Un ejemplo de script para crear un disco de arranque:

```
#!/bin/sh
#
```

```
# crear un disco de recuperacion
#
# formatear el disco
#
PATH=/bin:/sbin:/usr/sbin:/usr/bin

fdformat -q fd0
if [ $? -ne 0 ]
then
  echo "Disco defectuoso, por favor utilice uno nuevo."
  exit 1
fi

# escribir bloques de arranque en el disco
#
disklabel -w -B /dev/rfd0c fd1440

#
# crear un nuevo sistema de archivos en la unica particion existente
#
newfs -t 2 -u 18 -l 1 -c 40 -i 5120 -m 5 -o space /dev/rfd0a

#
# montar el nuevo disco
#
mount /dev/fd0a /mnt

#
# crear los directorios necesarios
#
mkdir /mnt/dev
mkdir /mnt/bin
mkdir /mnt/sbin
mkdir /mnt/etc
mkdir /mnt/root
mkdir /mnt/mnt # para la particion raiz
mkdir /mnt/tmp
mkdir /mnt/var

#
# poblar los directorios
#
if [ ! -x /sys/compile/MINI/kernel ]
then
  cat « EOM
```

```
El nucleo MINI no existe, por favor cree uno.
Aqui tiene un ejemplo de archivo de configuracion:
#
# MINI - Un nucleo para poner FreeBSD en un disco
#
machine "i386"
cpu "I486_CPU"
ident MINI
maxusers 5

options INET # necesario para _tcp _icmpstat _ipstat
  # _udpstat _tcpstat _udb
options FFS # sistema de archivos rapido (Fast
                                        # File System) de Berkeley
options FAT_CURSOR # cursor en forma de bloque en syscons o
                                        # pccons
options SCSI_DELAY=15 # ser pesimista acerca de dispositivos
                                        # SCSI genericos
options NCONS=2         # 1 consola virtual
options USERCONFIG # permitir la configuracion por el
                                        # usuario mediante -c XXX

config kernel root on da0 swap on da0 and da1 dumps on da0

controller isa0
controller pci0

controller fdc0 at isa? port "IO_FD1" bio irq 6 drq 2 vector fdintr
disk fd0 at fdc0 drive 0

controller ncr0

controller scbus0

device sc0 at isa? port "IO_KBD" tty irq 1 vector scintr
device npx0 at isa? port "IO_NPX" irq 13 vector npxintr

device da0
device da1
device da2

device sa0

pseudo-device loop # requerido por INET
pseudo-device gzip # ejecutar archivos en formato a.out que estan
```

```
                                # comprimidos con gzip
EOM
  exit 1
fi

cp -f /sys/compile/MINI/kernel /mnt

gzip -c -best /sbin/init > /mnt/sbin/init
gzip -c -best /sbin/fsck > /mnt/sbin/fsck
gzip -c -best /sbin/mount > /mnt/sbin/mount
gzip -c -best /sbin/halt > /mnt/sbin/halt
gzip -c -best /sbin/restore > /mnt/sbin/restore

gzip -c -best /bin/sh > /mnt/bin/sh
gzip -c -best /bin/sync > /mnt/bin/sync

cp /root/.profile /mnt/root

cp -f /dev/MAKEDEV /mnt/dev
chmod 755 /mnt/dev/MAKEDEV

chmod 500 /mnt/sbin/init
chmod 555 /mnt/sbin/fsck /mnt/sbin/mount /mnt/sbin/halt
chmod 555 /mnt/bin/sh /mnt/bin/sync
chmod 6555 /mnt/sbin/restore

#
# crear los nodos de los dispositivos
#
cd /mnt/dev
./MAKEDEV std
./MAKEDEV da0
./MAKEDEV da1
./MAKEDEV da2
./MAKEDEV sa0
./MAKEDEV pty0
cd /

#
# crear una tabla minima de sistemas de archivos
#
cat > /mnt/etc/fstab «EOM
/dev/fd0a / ufs rw 1 1
EOM
```

```
#
# crear un archivo de passwords minimo
#
cat > /mnt/etc/passwd «EOM
root:*:0:0:Charlie &:/root:/bin/sh
EOM

cat > /mnt/etc/master.passwd «EOM
root::0:0::0:0:Charlie &:/root:/bin/sh
EOM

chmod 600 /mnt/etc/master.passwd
chmod 644 /mnt/etc/passwd
/usr/sbin/pwd_mkdb -d/mnt/etc /mnt/etc/master.passwd

#
# desmontar el disco e informar al usuario
#
/sbin/umount /mnt
```

9.2.8.2. Tras el desastre

La pregunta clave es: ¿ha sobrevivido su hardware? Como ha estado usted realizando copias de seguridad regularmente, no hace falta que se preocupe por sus datos.

Si se ha danado su hardware, comience por sustituir aquellos dispositivos que se hayan estropeado.

Si su hardware se encuentra bien, compruebe sus discos. Si está usando un disco de arranque creado a medida, arranque en modo monousuario (teclee `-s` cuando aparezca el prompt `boot:`). Sáltese el siguiente párrafo.

Si está usando los disquetes `boot.flp` y `fixit.flp`, continúe leyendo. Introduzca el disquete `boot.flp` en la primera unidad e inicie el ordenador. En la pantalla aparecerá el menú original de instalación. Seleccione la opción de reparación, `Fixit-Repair mode with CDROM or floppy`. Cuando se le solicite, introduzca el disquete `fixit.flp`. `restore` y los otros programas que necesita se encuentran en `/mnt2/stand`.

Recupere cada sistema de archivos por separado.

Intente montar la partición raíz de su primer disco mediante mount(8) (por ejemplo, `mount /dev/da0a /mnt`). Si la tabla de particiones BSD (disklabel) se encuentra danada, use disklabel(8) para reparticionar y etiquetar el disco de modo que coincida con la etiqueta que usted imprimió y guardó. Utilice newfs(8) para volver a crear los sistemas de archivos. Vuelva a montar la partición raíz del disquete en modo de lectura y escritura (`mount -u -o rw /mnt`). Use los programas apropiados y las cintas que contienen las copias de seguridad para recuperar los datos de este sistema de archivos (por ejemplo, `restore vrf /dev/sa0`). Desmonte el sistema de archivos (por ejemplo, `umount /mnt`). Repita este proceso para cada sistema de archivos que resultase danado.

Una vez que su sistema se encuentre funcionando, copie sus datos en cintas nuevas. Sea lo que fuere lo que causó la pérdida de datos, puede suceder nuevamente. Emplear una hora en esta tarea le puede salvar de problemas adicionales en el futuro.

9.2.8.3. * No me preparé para el desastre, ¿Qué hago ahora?

Capítulo 10. Cuotas de disco

Contribuido por Mike Pritchard <mpp@FreeBSD.org>. 26 Febrero 1996

Las cuotas constituyen una prestación del sistema operativo que le permite limitar la cantidad de espacio en disco y/o el número de archivos de que un usuario puede disponer en un determinado sistema de archivos. Habitualmente se utiliza en sistemas multiusuario en los que resulta deseable limitar la cantidad de recursos de que puede disponer cada usuario o grupo de usuarios. De este modo se impide que un usuario agote todo el espacio disponible en un disco.

10.1. Configurando su sistema para habilitar las cuotas de disco

Antes de intentar usar las cuotas de disco es necesario asegurarse de que se encuentran habilitadas en el núcleo del sistema operativo. Esto se consigue anadiendo la siguiente línea al archivo de configuración del núcleo:

```
options QUOTA
```

El núcleo que se incluye de serie (`GENERIC`) no tiene esta función activada por defecto, por lo que tendrá que configurar, crear e instalar un núcleo a medida para utilizar las cuotas de disco. Puede acudir a la sección Configurando el núcleo de FreeBSD, donde encontrará más información acerca de la configuración del núcleo.

A continuación, necesitará habilitar las cuotas de disco en `/etc/sysconfig`. Esto se hace cambiando la línea:

```
quotas=NO
```

por:

```
quotas=YES
```

Si está utilizando FreeBSD 2.2.2 o posterior, el archivo de configuración será `/etc/rc.conf` y el nombre de la variable habrá cambiado a:

```
check_quotas=YES
```

Finalmente, necesitará editar `/etc/fstab` para habilitar las cuotas de disco de forma individualizada para cada sistema de archivos. Es aquí donde puede habilitar cuotas para usuarios, grupos o ambos y para todos sus sistemas de archivos.

Para habilitar las cuotas de disco para un determinado usuario en un sistema de archivos, anada la opción `userquota` en el archivo `/etc/fstab`, en el campo de opciones del registro correspondiente al sistema de archivos para el cual quiere habilitar las cuotas. Por ejemplo:

```
/dev/da1s2g   /home    ufs rw,userquota 1 2
```

Análogamente, para habilitar las cuotas para grupos emplee la opción `groupquota` en lugar de `userquota`. Para habilitar ambas, cuotas de usuario y de grupo, cambie el registro del siguiente modo:

```
/dev/da1s2g    /home    ufs rw,userquota,groupquota 1 2
```

Por defecto, los archivos relacionados con las cuotas se almacenan en el directorio raíz del sistema de archivos con los nombres `quota.user` y `quota.group`, para las cuotas de usuarios y de grupos, respectivamente. Consulte `man fstab` para más información. Aunque esa página man afirma que se puede especificar un emplazamiento alternativo para los archivos relacionados con las cuotas, esto no es recomendable ya que las diversas utilidades de las cuotas no parecen adaptarse convenientemente a este cambio.

En este momento debería reiniciar su sistema con el nuevo núcleo. `/etc/rc` ejecutará automáticamente los comandos apropiados para crear los archivos iniciales de las cuotas para todas las cuotas que haya activado en `/etc/fstab`, por lo que no hace falta crear manualmente ningún archivo de cuotas de logitud cero.

Durante el funcionamiento normal del sistema, no debería ser necesario ejecutar manualmente los comandos `quotacheck`, `quotaon`, o `quotaoff`. Sin embargo, puede interesarle leer sus páginas man para familiarizarse con su funcionamiento.

10.2. Fijando los límites de las cuotas

Una vez haya configurado su sistema para habilitar las cuotas, compruebe que realmente han sido habilitadas. Una forma de hacerlo fácilmente es ejecutar

```
# quota -v
```

Debería aparecer una línea describiendo la utilización de los discos y los límites actuales de cuotas para cada sistema de archivos en el que estén activadas las cuotas.

Ya está listo para comenzar a asignar límites de cuota con el mandato `edquota`.

Tiene varias opciones acerca de cómo aplicar límites a la cantidad de espacio en disco de que un usuario o grupo puede disponer y la cantidad de archivos que pueden crear. Puede limitar la cantidad de disco utilizada en función del espacio físico usado (cuotas de bloque), en función de los archivos (cuotas de inode) o en función de una combinación de ambos. Cada uno de estos límites se divide en dos categorías: límites flexibles y rígidos.

Los límites rígidos no pueden ser sobrepasados. Cuando un usuario alcanza su límite rígido, ya no puede disponer de más espacio en el sistema de archivos en cuestión. Por ejemplo, si el usuario tiene un límite rígido de 500 bloques en

un determinado sistema de archivos y ya está utilizando 490 bloques, el usuario sólo puede asignar 10 bloques adicionales. Cualquier intento de asignar 11 bloques fracasará.

Por otra parte, los límites flexibles pueden ser sobrepasados por un periodo de tiempo limitado. Este periodo de tiempo se conoce como el periodo de gracia, que dura una semana por defecto. Si un usuario permanece por encima de su límite flexible más allá del periodo de gracia, el límite flexible se convertirá en un límite rígido y no se permitirá realizar nuevas asignaciones. Cuando el usuario se vuelva a situar por debajo del límite flexible, el periodo de gracia se reiniciará.

A continuación se muestra un ejemplo de lo que podrá ver cuando ejecute el comando `edquota`. Cuando se invoca el comando `edquota`, se inicia el editor especificado en la variable de entorno EDITOR, o el editor `vi` en caso de que la variable EDITOR no se encuentre definida, permitiéndole editar los límites de las cuotas de disco.

```
# edquota -u test

Quotas for user test:
/usr: blocks in use: 65, limits (soft = 50, hard = 75)
        inodes in use: 7, limits (soft = 50, hard = 60)
/usr/var: blocks in use: 0, limits (soft = 50, hard = 75)
        inodes in use: 0, limits (soft = 50, hard = 60)
```

Normalmente verá dos líneas por cada sistema de archivos que tenga activadas las cuotas de disco. Una línea para los límites de bloques y otra para los límites de inodes. Para modificar los límites de las cuotas, simplemente cambie el valor que quiere actualizar. Por ejemplo, para elevar el límite de bloques de este usuario de un límite flexible de 50 y un límite rígido de 75 a un límite flexible de 500 y a un límite rígido de 600, cambie:

```
/usr: blocks in use: 65, limits (soft = 50, hard = 75)
```

por:

```
/usr: blocks in use: 65, limits (soft = 500, hard = 600)
```

Los nuevos límites estarán vigentes cuando salga del editor.

En ocasiones resulta deseable imponer límites en las cuotas de disco a un intervalo de UIDs (número de identificación de usuario). Esto se puede hacer empleando la opción `-p` del comando `edquota`. En primer lugar, asigne el límite deseado al usuario y luego ejecute `edquota -p protouser UID_inicial-UID_final`. Por ejemplo, si el usuario `test` está sujeto a límites en la cuota de disco, el siguiente comando puede ser empleado para duplicar esos límites para los UIDs comprendidos entre 10000 y 19999:

```
# edquota -p test 10000-19999
```

La capacidad de especificar intervalos de UIDs se anadió al sistema después de que la versión 2.1 fuese distribuida. Si necesita esta función en un sistema 2.1, deberá conseguir una copia más reciente de edquota.

Consulte `man edquota`, donde encontrará información más detallada.

10.3. Comprobando los límites en las cuotas y la utilización de los discos

Puede utilizar el comando `quota`, o bien el comando `repquota` para comprobar los límites en las cuotas y la utilización de los discos. El comando `quota` puede ser empleado para comprobar cuotas individuales y de grupo, así como la utilización de los discos. Sólo el administrador del sistema puede examinar las cuotas y la utilización de los discos por otros usuarios o por grupos de los que no es miembro. El comando `repquota` puede ser empleado para obtener un resumen de todas las cuotas y la utilización de los discos para sistemas de archivos que tengan las cuotas habilitadas.

A continuación se muestra un ejemplo de la salida del comando `quota -v` para un usuario que está sujeto a límites en la cuota de disco en dos sistemas de archivos.

```
Disk quotas for user test (uid 1002):
     Filesystem  blocks   quota   limit   grace   files   quota   limit   grace
           /usr      65*     50      75   5days       7      50      60
       /usr/var       0      50      75               0      50      60
```

En el ejemplo anterior, en el sistema de archivos `/usr` el usuario ya ha sobrepasado en 15 bloques su límite flexible (fijado en 50 bloques) y aún dispone de 5 días del periodo de gracia. Observe la presencia de un asterisco `*`, lo cual indica que el usuario ha sobrepasado su límite en la cuota de disco.

Normalmente, aquellos sistemas de archivos en los que el usuario no esté utilizando espacio no figurarán en la salida del comando `quota`, aunque el usuario tenga asignado un límite en la cuota de disco para esos sistemas de archivos. La opción `-v` mostrará esos sistemas de archivos, como ha sucedido con `/usr/var` en el ejemplo anterior.

10.4. * Cuotas a través de NFS

Esta sección se encuentra en desarrollo.

Capítulo 11. El sistema X Window

El sistema X Window

Capítulo 12. Compatibilidad de Hardware

Compatibilidad de Hardware

Capítulo 13. Localizacion

Localizacion

III. Comunicaciones en Red

Capítulo 14. Comunicaciones Serie

Comunicaciones Serie

Capítulo 15. PPP y SLIP

PPP y SLIP

15.1. User-PPP

Capítulo 16. Networking Avanzado

Networking Avanzado

Capítulo 17. Correo Electrónico

Correo Electrónico

IV. Conceptos Avanzados

Capítulo 18. FreBSD-current y FreeBSD-stable

FreBSD-current y FreeBSD-stable

18.1. Mantenerse -current con FreeBSD

18.2. Mantenerse -stable con FreeBSD

18.3. Sincronizando el código fuente a través de Internet

Capítulo 19. Contribuyendo a FreeBSD

Contribuyendo a FreeBSD

19.1. Contribuyentes adicionales de FreeBSD

Capítulo 20. Guías y políticas del árbol de código fuente

Contribución de Poul-Henning Kamp <phk@FreeBSD.org>.

Este capítulo documenta la política por la que se rige el árbol de código fuente de FreeBSD.

20.1. `MAINTAINER` en Makefiles

Junio 1996.

Si una parte en particular de la distribución FreeBSD es mantenida por una persona o grupo de personas, pueden comunicar este hecho al mundo anadiendo esta línea

```
MAINTAINER= email-addresses
```

a los `Makefile`s que cubren esta parte del árbol de código fuente.

La semántica de es como sigue:

El mantenedor posee y es responsable de ese código. Esto significa que él es responsable de arreglar errores y responder reportes de problemas relacionados con esa parte del código, y en el caso de contribución de software, de revisar nuevas versiones, según sea necesario.

Los cambios en directorios que tienen un mantenedor definido deberán ser enviados al mantenedor para ser revisados antes de ser incluidos. Sólo si el mantenedor no responde durante un periodo de tiempo inaceptable, a varios mensajes, será aceptable incluir cambios sin la revisión del mantenedor. Sin embargo, se sugiere que los cambios sean revisados por alguien más si es posible.

No es aceptable, por supuesto, anadir una persona o grupo como mantenedor a menos que estén de acuerdo en asumir esta tarea. Por otro lado, no tiene porqué ser un committer, pudiendo ser fácilmente un grupo de personas.

20.2. Software de Contribución

Contribucion de Poul-Henning Kamp <phk@FreeBSD.org> y David O'Brien <obrien@FreeBSD.org>.

Junio 1996.

Algunas partes de la distribución de FreeBSD consisten en software que está siendo mantenido activamente fuera del proyecto FreeBSD. Por razones históricas, llamamos a esto software *de contribución*. Algunos ejemplos son perl, gcc y patch.

En los últimos dos anos, se han usado diferentes métodos para tratar con este tipo de software y todos tienen algunas ventajas e inconvenientes. No ha emergido ningún claro ganador.

Ya que este es el caso, después de algo de debate uno de estos métodos ha sido seleccionado como el método “oficial” y será requerido para futuras importaciones de este tipo de software. Más aún, se sugiere firmemente que el software de contribución existente converja con este modelo en el futuro, ya que tiene ventajas significativas sobre el antiguo método, incluyendo la habilidad de obtener fácilmente diferencias relativas a la versión “oficial” del fuente por todos (aún sin acceso cvs). Esto hará significativamente mas fácil el devolver cambios a los desarrolladores primarios del software de contribución.

Finalmente, sin embargo, quedan las personas que están haciendo el trabajo. Si usar este modelo es particularmente incompatible con el paquete con el cual se trata, se pueden conceder excepciones a esta regla solo con la aprobación del core team y con el consenso general de los otros desarrolladores. La habilidad de mantener el paquete en el futuro será un asunto clave en las decisiones.

> **Nota:** Debido a algunas desafortunadas limitaciones de diseño con el formato de archivo RCS y el uso de las ramas "vendors" con el CVS, son *fuertemente desaprobados* cambios cosméticos, triviales y/o menores en archivos que estén siendo seguidos por la rama "vendor". Los "arreglos de sintaxis" están incluidos explícitamente aquí bajo la categoría "cosméticos" y deben ser evitados en archivos con revisión 1.1.x.x. El impacto que puede tener la modificación de un sólo carácter en el repositorio puede ser bastante dramático.

El lenguaje de programación **Tcl** será usado como un ejemplo sobre como este modelo trabaja:

`src/contrib/tcl` contiene los fuentes distribuidos por los mantenedores de este programa. Las partes que no son enteramente aplicables a FreeBSD pueden ser eliminadas. En el caso del Tcl, los subdirectorios `mac`, `win` y `compat` fueron eliminados antes de la importación

`src/lib/libtcl` contiene sólo un `Makefile` estilo bmake que usa las reglas estándar `bsd.lib.mk` makefile para producir la librería e instalar la documentación.

`src/usr.bin/tclsh` contiene sólo un `Makefile` estilo bmake que producirá e instalará el programa `tclsh` y sus páginas man asociadas usando las reglas estándar `bsd.prog.mk`

`src/tools/tools/tcl_bmake` contiene un par de shell scripts que pueden ser de ayuda cuando el software tcl necesita actualización. Estos no son parte integral o de instalación del software.

Lo importante aquí es que el directorio `src/contrib/tcl` se crea de acuerdo a las reglas: se supone que debe contener las fuentes tal y como se distribuyen (en una rama vendor adecuada del CVS y sin expansión de claves RCS) con tan pocos cambios específicos para FreeBSD como sea posible. La herramienta "easy-import"(importación fácil) en freefall ayudará haciendo la importación, pero si hay dudas acerca de como hacerlo, es imperativo que se pregunte primero y no hacerlo esperando que “funcione”. El CVS no perdona accidentes de importación y se requiere un gran esfuerzo para arreglar grandes errores.

Debido a las limitaciones de diseno previamente mencionadas de las ramas "vendor"del CVS, se requiere que los parches “oficiales” del vendedor sean aplicados a los fuentes originales de distribución y el resultado reimportado otra vez en la rama vendor correspondiente. Los parches oficiales nunca deben ser usados en la versión disponible en el CVS de FreeBSD e integrados, ya que esto destruye la coherencia de la rama vendor y hace que la importación de futuras versiones sea más complicada por la aparición de conflictos.

Ya que muchos paquetes contienen archivos cuya intención es la compatibilidad con otras arquitecturas y ambientes distintos a FreeBSD, es permisible eliminar partes del árbol de distribución que no son de interés para FreeBSD con el fin de ahorrar espacio. Los archivos que contienen notas de copyright y notas de la versión, información en algo aplicable a los archivos que quedan *no* deberán ser eliminados.

Si parece más fácil, los `bmake` `Makefile`s pueden ser producidos automaticamente desde el árbol de distribución por alguna utilidad, algo que podría hacer aun más fácil el actualizar a una nueva versión. Si esto es hecho, asegúrese de revisar tales utilidades (como sea necesario) en el directorio `src/tools` junto con el mismo port para que esté disponible para los futuros mantenedores.

En el nivel de directorio `src/contrib/tcl` debe ser andido un archivo llamado `FREEBSD-upgrade` debiendo presentar cosas como:

- Qué archivos se han dejado fuera de la importación
- De donde fue obtenida la distribución original y/o el servidor oficial principal.
- Dónde enviar parches a los autores originales
- Una visión general de los cambios específicos que se han hecho para FreeBSD.

Sin embargo, por favor no importar `FREEBSD-upgrade (actualización)` con los fuentes de contribución. En su lugar, usar `cvs add FREEBSD-upgrade ; cvs ci` después del prompt inicial. Más abajo existe un ejemplo de sintaxis de `src/contrib/cpio`:

```
Este directorio contiene códigos fuente virgen de los archivos originales de
distribución en una rama "vendor". No tratar, bajo ninguna circunstancia,
de actualizar los archivos en este directorio con parches y un cvs commit.
Nuevas versiones o versiones oficiales de parches deben ser importadas.
Por favor recuerdar importar con "-ko" para prevenir que el CVS corrompa
el Id del RCS de algun vendor.

Para la importación del GNU cpio 2.4.2, se eliminaron los siguientes archivos:

        INSTALL         cpio.info       mkdir.c
        Makefile.in     cpio.texi       mkinstalldirs

Para actualizar a una versión más nueva de cpio, cuando esté disponible:
        1. Descomprimir la nueva versión en un directorio vacío.
           [No hacer NINGUN cambio a los archivos.]
```

```
        2. Eliminar los archivos listados arriba y cualquier otro que no se relacione
           con FreeBSD.

        3. Usar el comando:
                cvs import -ko -m 'Virgin import of GNU cpio v<version>' \
                        src/contrib/cpio GNU cpio_<version>

           Por ejemplo, para importar la versión 2.4.2, usar:
                cvs import -ko -m 'Virgin import of GNU v2.4.2' \
                        src/contrib/cpio GNU cpio_2_4_2

        4. Seguir las instrucciones del paso 3 para resolver cualquier conflicto entre
           cambios locales de FreeBSD y la nueva versión.

NO desviarse, bajo ninguna circunstancia, de este procedimiento.

Para hacer cambios locales al cpio, simplemente aplicar el patch y commit a la rama
principal (también conocida como HEAD).  Nunca hacer cambios locales
en la rama GNU.

Todos los cambios locales deberían ser enviados a "cpio@gnu.ai.mit.edu" para
su inclusión en la nueva versión release.

obrien@FreeBSD.org - 30 March 1997
```

20.3. Archivos "adicionales" (encumbered)

Ocasionalmente podría ser necesario incluir un archivo "adicional"en el código fuente de FreeBSD. Por ejemplo, si un dispositivo requiere que una pequena pieza de código binario sea cargada antes de que el dispositivo funcione, y no tenemos los fuentes de ese código, entonces se dice que el archivo binario es "encumbered". Las siguientes políticas se aplican al incluir archivos encumbered en en el árbol fuente FreeBSD.

1. Cualquier archivo que sea interpretado o ejecutado por la CPU del sistema y no en formato fuente es encumbered.
2. Cualquier archivo con una licencia más restrictiva que BSD o GNU es encumbered.
3. Un archivo que contiene datos binarios descargables para que el hardware lo use no es encumbered, a menos que (1) o (2) se apliquen a él. Debe ser guardado en un formato ASCII neutral a la arquitectura (se recomienda file2c o uuencodeado).

4. Cualquier archivo encumbered requiere aprobación específica del Core team antes de ser añadido al repositorio CVS.
5. Los archivos encumbered van en `src/contrib` o `src/sys/contrib`.
6. El módulo entero debería ser mantenido en su conjunto. No hay excusa en separarlo, a menos que haya intercambio de código con código no encumbered.
7. Los archivos objeto son nombrados *`arch/filename`*`.o.uu>` .
8. Archivos del Kernel:
    a. Siempre deben ser referenciados en `conf/files.*` (para simplicidad del build).
    b. Siempre debería estar en `LINT`, pero el Core team decide caso por caso si debería ser comentado o no. El Core team puede, por supuesto, cambiar su opinión más adelante.
    c. El Ingeniero de Release decide si va o no a la release.
9. Archivos de usuario:
    a. El Core team decide si el código debería ser parte de `make world`.
    b. El Ingeniero de la Release decide si va a la release.

20.4. Librerías compartidas

Contribuido por Satoshi Asami <`asami@FreeBSD.org`>, Peter Wemm <`peter@FreeBSD.org`>, y David O'Brien <`obrien@FreeBSD.org`> 9 Diciembre 1996.

Si se está anadiendo soporte para librerías compartidas a un puerto u otra pieza de software que no tiene uno, los números de versión deben seguir estas reglas. Generalmente, los números resultantes no tendrán nada que ver con la versión release del software.

Los tres principios de la construcción de librerías compartidas son:

- Comenzando desde `1.0`
- Si hay un cambio que sea compatible con versiones anteriores, eliminar el número menor
- Si hay un cambio incompatible, quitar el número mayor

Por ejemplo, funciones anadidas y solució de errores resultan en la eliminación del numero de versión menor, mientras que las funciones borradas, sintaxis cambiada de llamada de función, etc, forzarán que el número de versión mayor cambie.

Usar los números de versión de la forma mayor.menor (*x*.*y*). Nuestro lincador dinámico no gestiona correctamente números de versión de la forma *x*.*y*.*z* . Cualquier número de versión después de *y* (es decir, el tercer dígito) es totalmente ignorado cuando se comparan números de versión de librerías compartidas para decidir con que librería enlazar. Dadas 2 librerías compartidas que difieren sólo en la “micro” revisión, `ld.so` enlazará con la mas alta. Es decir: si se enlaza con `libfoo.so.3.3.3`, el lincador sólo reconoce `3.3` en los encabezados, y enlazará con cualquiera que comience con *libfoo.so.3*.*(cualquier cosa >=3)*.*(el más alto disponible)*.

> **Nota:** `ld.so` siempre usará la revisión “menor” más alta. Es decir: usará `libc.so.2.2` en preferencia a `libc.so.2.0`, aun si el programa estaba inicialmente enlazado con `libc.so.2.0`.

Para librerías no portables, es también nuestra política cambiar el número de versión de librería compartida solo una vez entre releases. Cuando se hace un cambio a una librería del sistema que requiere que se quite el número de versión, revisar los logs commit del `Makefile`. Es responsabilidad del miembro del commit asegurarse de que el primer cambio desde la release hará que se actualice el número de versión de la librería compartida en el `Makefile`, y cualquier subsecuente cambio no lo hará.

Capítulo 21. Añadir nuevas opciones de configuración al kernel

Contribuci´n de Jörg Wunsch <joerg@FreeBSD.org>

> **Nota:** Antes de leer esta sección se debe estar familiarizado con la sección sobre configuración del kernel.

21.1. Ante todo ¿Qué es una *opción del kernel*?

El uso de opciones del kernel se describe básicamente en la sección configuración del kernel. También hay una discusión sobre opciones “estilo historico” y “estilo moderno”. El objetivo final es que en algún momento todas las opciones soportadas por el kernel se hayan convertido al estilo moderno, así para la gente que hizo un `make depend` exitoso en su directorio de compilación del kernel luego de correr config(8), el proceso de build automaticamente seleccionará las opciones modificadas y solo recompilara los archivos donde se usan. El paso de eliminar el viejo directorio de compilación cada vez que se corre config(8) como aún se hace podrá entonces ser eliminado nuevamente.

Básicamente una opción del kernel no es más que la definición de una macro del preprocesador C para el proceso de compilación del kernel. Para hacer la compilación verdaderamente opcional, la parte correspondiente del código fuente del kernel (o archivo `.h` del kernel) debe estar escrita con el concepto de opción en mente, por ejemplo el default para la opción debe haber sido hecho modificable. Esto se hace generalmente con algo como:

```
#ifndef THIS_OPTION
#define THIS_OPTION (valor_por_default)
#endif /* THIS_OPTION */
```

De esta manera, un administrador seleccionando otro valor para la opción en su archivo de configuración dejará sin efecto el default, y lo reemplazará con su nuevo valor. Como es claro, el nuevo valor será sustituido en el código fuente durante la ejecución del preprocesador, por lo que debe ser una expresión C válida en el contexto en el que habría sido usado el default.

También es posible crear opciones sin valor que simplemente activen o desactiven una parte del código rodeandola con

```
#ifdef THAT_OPTION

[código fuente específico]

#endif
```

Con solo mencionar `THAT_OPTION` en el archivo de configuración (con o sin valor) activará la parte del código correspondiente.

Quien esté familiarizado con el lenguaje C se dará cuenta de que todo podría ser tratado como una “opción de configuración” si hay por lo menos un `#ifdef` referenciandolo... De todos modos, es poco probable que mucha gente ponga

```
options notyet,notdef
```

en su archivo de configuración, y luego se pregunten por que la kernel compilation les da errores. :-)

Como es claro, usar nombres arbitrarios para las opciones hace que sea muy difícil rastrear su uso por todo el árbol de código fuente. Esa es la razón que impulsa el esquema de opciones *estilo-moderno*, donde cada opción va en un archivo `.h` separado en el directorio de compilación del kernel, el cual por convención se llamará `opt_`*`foo`*`.h`. De esta forma las dependencias usuales del Makefile pueden aplicarse, y el `make` puede determinar que se necesita recompilar cuando una opción fue cambiada.

El mecanismo de opciones estilo antiguo tiene aún una ventaja para opciones locales o tal vez experimentales que tendrán por anticipado poco tiempo de vida: ya que es facil agregar un nuevo `#ifdef` al código fuente del kernel, esto ya lo ha convertido en una opción de configuración. En este caso el administrador que use tal opción es responsable de conocer sus implicaciones (y tal vez forzar la recompilación de partes del kernel a mano). Una vez que la tansición de todas las opciones soportadas haya sido hecha, config(8) advertirá cuando aparezca una opción no soportada en el archivo de configuración, pero sin embargo la incluirá en el Makefile del kernel.

21.2. Y ahora ¿Qué debo hacer para eso?

Primero, edite `sys/conf/options` (o `sys/i386/conf/options.`<*`arq`*>, Ej: `sys/i386/conf/options.i386`), y seleccione un archivo `opt_`*`foo`*`.h` donde su nueva opción cabría mejor.

Si ya hay algo que se acerque al proposito de la nueva opción, elijalo. Por ejemplo, las opciones que modifican la conducta general del subsistema SCSI pueden ir dentro de `opt_scsi.h`. Por default, con solo mencionar una opción en el archivo de opciones apropiado, digamos `FOO`, implica que su valor irá dentro del archivo correspondiente `opt_foo.h`. Esto puede ser reemplazado en el lado derecho de una regla especificando otro nombre de archivo.

Si no hay ningún `opt_`*`foo`*`.h` ya disponible para la nueva opción, invente un nuevo nombre. Hágalo significativo, y comente la nueva sección en el archivo `options[.`<*`arq`*>`]`. config(8) automaticamente tomará el cambio, y creará el archivo la próxima vez que se corra. La mayoría de las opciones deberían ir en un archivo `.h` individual.

Meter demasiadas opciones en un solo archivo `opt_`*`foo`*`.h` causará que se tengan que recompilar demasiados archivos del kernel cuando se cambie una de las opciones en el archivo de configuración.

Por último, averigüe que archivos del kernel dependen de la nueva opción. A menos que usted haya inventado recientemente su opción, y aún no existe en ningún lado

```
  %
        find /usr/src/sys -name
type f | xargs fgrep NEW_OPTION
```

le será de ayuda para encontrarlos. Edite todos esos archivos y agregue

```
#include "opt_foo.h"
```

al comienzo, antes de todos los `#include <xxx.h>`. Este orden es más importante en tanto las opciones podrían reemplazar los defaults de los archivos `.h` regulares, si los defaults son de la forma

```
#ifndef NEW_OPTION #define NEW_OPTION (something)
#endif
```

en el `.h` regular.

Agregar una opción que reemplaza algo en un archivo header del sistema (un archivo ubicado en `/usr/include/sys/`) es casi siempre un error. `opt_`*`foo`*`.h` no puede ser incluido dentro de estos archivos dado que esto danaría al header muy seriamente, pero si no se incluye entonces otros lugares que lo incluyen podrían tener valores inconsistentes para la opción. Sí, hay precedentes de esto actualmente, pero eso no los hace más correctos.

Capítulo 22. Depurando el Kernel

Contribución de Paul Richards <paul@FreeBSD.org> y Jörg Wunsch <joerg@FreeBSD.org>

22.1. Depuración de un Kernel Crash Dump con kgdb

Aquí se dan instrucciones para hacer funcionar la depuración del kernel sobre un crash dump. En ellos se asume que usted tiene suficiente espacio de swap para un crash dump. Si usted tiene varias particiones de swap y la primera es demasiado pequena para albergar un dump, puede configurar su kernel para que use un dispositivo alternativo (en la línea `config kernel=`, o puede especificar una alternativa usando el comando dumpon(8). La mejor manera de usar dumpon(8) es asignar la variable `dumpdev` en `/etc/rc.conf`. Generalmente usted va a querer especificar uno de los dispositivos de swap especificados en `/etc/fstab`. Actualmente no se soportan dumps a dispositivos que no sean de swap, como por ejemplo una unidad de cinta. Configure su kernel usando `config -g`. Vea Configuración del Kernel para más detalles sobre la configuración del kernel FreeBSD.

Use el comando dumpon(8) para decirle al kernel donde hacer el dump (Tenga en cuenta que esto debera hacerse luego de configurar la partición como dispositivo de swap usando swapon(8)). Esto se hace normalmente usando los archivos `/etc/rc.conf` y `/etc/rc`. Como alternativa se puede fijar el dispositivo usando la cláusula `dump` en la línea `config` de su archivo de configuración del kernel. Esta práctica esta en desuso y sólo debería ser usada para obtener un dump de un kernel que falla durante el inicio.

> **Nota:** En adelante, el término `kgdb` se referirá al `gdb` corriendo en “kernel debug mode”. Esto puede lograrse ya sea iniciando al `gdb` con la opción `-k`, o enlazandolo y arrancandolo con el nombre `kgdb`. Esto no se hace por default, y la idea está basicamente en desuso ya que a la gente de GNU no le gusta que sus herramientas se comporten distinto cuando se las llama por otro nombre. Esta posibilidad podrá perfectamente ser descontinuada en futuras versiones.

Una vez que el kernel ha sido compilado haga una copia, digamos `kernel.debug`, y luego corra `strip -g` sobre el original. Instale el original normalmente. Usted tambien podría instalar el kernel sin hacer el `strip`, pero los tiempos de busqueda de algunos programas en la tabla de símbolos aumentarán sensiblemente, y dado que el kernel se carga completo en memoria durante el inicio y no puede intercambiarse a disco luego, se desperdiciarán varios megabytes de memoria.

Si usted esta probando un nuevo kernel, por ejemplo tipeando el nombre del nuevo kernel en el prompt de inicio, pero necesita arrancar otro para tener su sistema funcionando de nuevo, arranquelo solo en modo monousuario usando la opción `-s` en el prompt de inicio , y luego siga los siguientes pasos:

```
# fsck -p
# mount -a -t ufs       # asi su file system para /var/crash se puede escribir
```

```
# savecore -N /kernel.panicked /var/crash
# exit                  # ...a modo multiusuario
```

Esto instruye a savecore(8) que use otro kernel para la extracción de nombres de símbolo. De otro modo usaria el kernel en uso y probablemente no haga nada ya que los símbolos del kernel y los del dump son diferentes.

Ahora, luego de un crash dump, vaya a `/sys/compile/WHATEVER` y corra `kgdb`. desde `kgdb` haga:

```
symbol-file kernel.debug
exec-file /var/crash/kernel.0
core-file /var/crash/vmcore.0
```

y voila, ya puede depurar el crash dump usando las fuentes del kernel como lo hace con cualquier otro programa.

A continuación se muestra el log de una sesión de `kgdb` que ilustra el procedimiento. Las líneas fueron numeradas para referencia, y las largas fueron cortadas para mejorar la legibilidad. Mas alla de esto es el registro de un error real tomado durante el desarrollo del driver pcvt de consola.

```
 1:Script started on Fri Dec 30 23:15:22 1994
 2:# cd /sys/compile/URIAH
 3:# kgdb kernel /var/crash/vmcore.1
 4:Reading symbol data from /usr/src/sys/compile/URIAH/kernel
...done.
 5:IdlePTD 1f3000
 6:panic: because you said to!
 7:current pcb at 1e3f70
 8:Reading in symbols for ../../i386/i386/machdep.c...done.
 9:(kgdb) where
10:#0  boot (arghowto=256) (../../i386/i386/machdep.c line 767)
11:#1  0xf0115159 in panic ()
12:#2  0xf01955bd in diediedie () (../../i386/i386/machdep.c line 698)
13:#3  0xf010185e in db_fncall ()
14:#4  0xf0101586 in db_command (-266509132, -266509516, -267381073)
15:#5  0xf0101711 in db_command_loop ()
16:#6  0xf01040a0 in db_trap ()
17:#7  0xf0192976 in kdb_trap (12, 0, -272630436, -266743723)
18:#8  0xf019d2eb in trap_fatal (...)
19:#9  0xf019ce60 in trap_pfault (...)
20:#10 0xf019cb2f in trap (...)
21:#11 0xf01932a1 in exception:calltrap ()
22:#12 0xf0191503 in cnopen (...)
23:#13 0xf0132c34 in spec_open ()
24:#14 0xf012d014 in vn_open ()
25:#15 0xf012a183 in open ()
26:#16 0xf019d4eb in syscall (...)
27:(kgdb) up 10
```

```
Reading in symbols for ../../i386/i386/trap.c...done.
#10 0xf019cb2f in trap (frame={tf_es = -260440048, tf_ds = 16, tf_\
edi = 3072, tf_esi = -266445372, tf_ebp = -272630356, tf_isp = -27\
2630396, tf_ebx = -266427884, tf_edx = 12, tf_ecx = -266427884, tf\
_eax = 64772224, tf_trapno = 12, tf_err = -272695296, tf_eip = -26\
6672343, tf_cs = -266469368, tf_eflags = 66066, tf_esp = 3072, tf_\
ss = -266427884}) (../../i386/i386/trap.c line 283)
283                             (void) trap_pfault(&frame, FALSE);
(kgdb) frame frame->tf_ebp frame->tf_eip
Reading in symbols for ../../i386/isa/pcvt/pcvt_drv.c...done.
#0  0xf01ae729 in pcopen (dev=3072, flag=3, mode=8192, p=(struct p\
roc *) 0xf07c0c00) (../../i386/isa/pcvt/pcvt_drv.c line 403)
403             return ((*linesw[tp->t_line].l_open)(dev, tp));
(kgdb) list
398
399             tp->t_state |= TS_CARR_ON;
400             tp->t_cflag |= CLOCAL;  /* cannot be a modem (:-) */
401
402     #if PCVT_NETBSD || (PCVT_FREEBSD >= 200)
403             return ((*linesw[tp->t_line].l_open)(dev, tp));
404     #else
405             return ((*linesw[tp->t_line].l_open)(dev, tp, flag));
406     #endif /* PCVT_NETBSD || (PCVT_FREEBSD >= 200) */
407     }
(kgdb) print tp
Reading in symbols for ../../i386/i386/cons.c...done.
$1 = (struct tty *) 0x1bae
(kgdb) print tp->t_line
$2 = 1767990816
(kgdb) up
#1  0xf0191503 in cnopen (dev=0x00000000, flag=3, mode=8192, p=(st\
ruct proc *) 0xf07c0c00) (../../i386/i386/cons.c line 126)
       return ((*cdevsw[major(dev)].d_open)(dev, flag, mode, p));
(kgdb) up
#2  0xf0132c34 in spec_open ()
(kgdb) up
#3  0xf012d014 in vn_open ()
(kgdb) up
#4  0xf012a183 in open ()
(kgdb) up
#5  0xf019d4eb in syscall (frame={tf_es = 39, tf_ds = 39, tf_edi =\
 2158592, tf_esi = 0, tf_ebp = -272638436, tf_isp = -272629788, tf\
_ebx = 7086, tf_edx = 1, tf_ecx = 0, tf_eax = 5, tf_trapno = 582, \
tf_err = 582, tf_eip = 75749, tf_cs = 31, tf_eflags = 582, tf_esp \
= -272638456, tf_ss = 39}) (../../i386/i386/trap.c line 673)
```

```
73:673             error = (*callp->sy_call)(p, args, rval);
74:(kgdb) up
75:Initial frame selected; you cannot go up.
76:(kgdb) quit
77:# exit
78:exit
79:
80:Script done on Fri Dec 30 23:18:04 1994
```

Comentarios al listado anterior:

línea 6:

Este es un dump tomado desde el DDB (vea abajo), a partir del comentario del panic “because you said to!”, y un listado de la pila bastante largo; la razón incial para entrar al DDB fue un trap de falta de pagina.

línea 20:

Esta es la ubicación de la función `trap()` en el listado de la pila.

línea 36:

Se fuerza el uso de un nuevo marco de pila; esto ya no es necesario ahora. Actualmente se supone que los marcos de pila apuntan a las ubicaciones correctas, aun en caso de un trap. (No tengo un dump más nuevo a mano <g>, mi kernel no ha hecho un panic desde hace bastante tiempo.) Mirando al código en la línea 403 del código fuente, hay una alta probabilidad de que la referencia al puntero “tp” sea erronea, o el acceso al arreglo estuviera fuera de sus límites.

línea 52:

El puntero se ve sospechoso, pero contiene una dirección válida.

línea 56:

De todos modos, obviamente apunta a basura, asi que encontramos nuestro error! (Para los que no esten familiarizados con esa pieza de código en particular: `tp->t_line` hace referencia a la disciplina de línea de este dispositivo de consola, el cual debe ser un entero por demás pequeño)

22.2. Depurando un crash dump con DDD

Tambien es posible examinar un crash dump del kernel con un debugger gráfico como `ddd`. Agregue la opción `-k` a la línea de comando del `ddd` que usaría normalmente. Por ejemplo;

```
# ddd -k /var/crash/kernel.0 /var/crash/vmcore.0
```

De esta manera usted debería poder analizar el crash dump usando la interfaz gráfica del `ddd`.

22.3. Analisis Post-mortem de un Dump

Que sucede si un kernel hace un dump de su memoria que usted no esperaba, y por lo tanto no estaba compilado usando `config -g`? No todo esta perdido aquí. Do not panic!

> **Nota:** N.del T.: El autor hace un juego con la palabra panic! que prefiero dejar sin traducir.

Por supuesto, aun necesita habilitar los crash dumps. Vea más arriba las opciones que debe especificar para esto.

Vaya a su directorio de configuración del kernel (`/usr/src/sys/`*`arq`*`/conf`) y edite su archivo de configuración. Quite las marcas de comentario (o agregue, si no existe) la siguiente línea

```
makeoptions    DEBUG=-g                #Build kernel with gdb(1) debug symbols
```

Recompile el kernel. Algunos otros archivos seran recompilados, por ejemplo `trap.o`, a causa del cambio de la fecha del archivo Makefile. Con un poco de suerte, la opción `-g` no cambiará nada del código generado, al final usted tendra un kernel con el mismo código que tiene problemas ahora pero con ciertos símbolos para depuración. Por lo menos usted debería verificar el tamano anterior y el nuevo con el comando size(1). Si hay diferencia probablemente sea el momento de darse por vencido.

Ahora puede examinar el dump como se describió anteriormente. Los símbolos pueden estar incompletos para algunas partes, como se puede ver en el listado de pila del ejemplo anterior donde algunas funciones se muestran sin numeros de línea ni listas de argumentos. Si necesita más símbolos borre los archivos objeto apropiados y repita la sesión de `kgdb` hasta que haya averiguado lo suficiente.

No se garantiza que todo esto funcione, pero irá bastante bien la en mayoría de los casos.

22.4. Depuración En-línea del Kernel Usando DDB

Si bien el `kgdb` provee un muy alto nivel de interfaz de usuario como depurador post-mortem, hay cosas que no puede hacer. Siendo las más importantes poder marcar puntos de interrupción y ejecutar código del kernel paso a paso.

Si usted necesita hacer una depuración a bajo nivel de su kernel hay disponible un debugger en-línea llamado DDB. Permite poner puntos de interrupción, ejecutar paso a paso las funciones del kernel, examinar y cambiar variables, etc. Sin embargo, no puede acceder al código fuente del kernel y sólo tiene acceso a los símbolos globales y estáticos, no a toda la información de depuración como el `kgdb`.

Para configurar su kernel para que incluya el DDB, agregue la opción

```
options DDB
```

a su archivo de configuración, y recompile (vea Configuración del Kernel para más detalles sobre como configurar el kernel de FreeBSD.)

> **Nota:** Notese que si usted tiene una versión vieja del boot block, sus símbolos de debugger podrían no ser cargados. Actualize su boot block; los mas recientes cargan los símbolos del DDB automágicamente.

Una vez que su kernel con DDB esta corriendo, hay varias maneras de entrar al DDB. La primera y más temprana es tipear la opción `-d` directamente en el prompt de inicio. El kernel se iniciará en modo de depuración e ingresará al DDB antes de cualquier detección de dispositivos. De aquí en adelante usted podrá depurar hasta las funciones probe/attach de los dispositivos.

El segundo escenario es una combinación de teclas, generalmente Ctrl-Alt-Esc. En el caso de las syscons, esto puede modificarse, algunos mapas de teclado distribuidos lo hacen, por lo que hay que prestar atención. Hay disponible una opción para consolas en puertos serie que permite el uso de la senal BREAK en la línea para entrar al DDB (`options BREAK_TO_DEBUGGER` en el archivo de configuración del kernel). Esto no es el default ya que hay un montón de adaptadores serie dando vueltas que generan condiciones BREAK sin necesidad, por ejemplo cuando se desenchufa el cable.

La tercera forma es que un panic salte al DDB si el kernel está configurado para usarlo. Por este motivo, no es recomendable configurar un kernel con DDB para una máquina funcionando sin atención.

Los comandos del DDB se asemejan remotamente a algunos de los del `gdb`. Lo primero que usted probablemente necesite hacer es poner un punto de interrupción:

```
b nombre-de-función
b dirección
```

Por default los números se toman en hexadecimal, pero para distinguirlos de los nombres de símbolo los números hexadecimales que empiezan con las letras `a-f` se deben preceder con `0x` (para los demás números esto es opcional). Tambien se admiten expresiones sencillas, por ejemplo: `nombre-de-función + 0x103`.

Para que el kernel interrumpido continue ejecutandose, solo tipee:

```
c
```

Para ver un listado de la pila, use:

```
trace
```

> **Nota:** Notese que cuando se entra al DDB con una combinación de teclas el kernel está atendiendo una interrupción, por lo que el listado de la pila podría no serle de mucha utilidad.

Si quiere quitar un punto de interrupción, use

```
del
del expresión-dirección-de-memoria
```

La primera forma se aceptará inmediatamente despues de llegar a un punto de interrupción, y borra el punto actual. La segunda puede quitar cualquier punto de interrupción, pero se debe especificar la dirección exacta; esta se puede obtener de:

```
show b
```

Para ejecutar el kernel paso a paso, intente:

```
s
```

Esto entrará dentro de las funciones, pero puede hacer que DDB las siga hasta llegar a la instrucción de retorno correspondiente usando:

```
n
```

> **Nota:** Esto es distinto de la instrucción `next` del `gdb`; es más parecido a la instrucción `finish`.

Para examinar los datos en la memoria use (por ejemplo):

```
x/wx 0xf0133fe0,40
x/hd db_symtab_space
x/bc termbuf,10
x/s stringbuf
```

para acceder a palabras/medias palabras/bytes, y para mostrar en hexadecimal/decimal/caracteres/strings. El número luego de la coma es la cantidad de objetos. Para mostrar los siguientes 0x10 items, simplemente use:

```
x ,10
```

Del mismo modo, use

```
x/ia foofunc,10
```

para desensamblar las primeras 0x10 instrucciones de `foofunc`, y mostrarlas junto con su desplazamiento desde el comienzo de `foofunc`.

Para modificar memoria, use el comando write:

```
w/b termbuf 0xa 0xb 0
w/w 0xf0010030 0 0
```

El modificador (`b`/`h`/`w`) especifica el tamano de los datos a ser escritos, la primera expresión a continuación es la dirección donde escribir y el resto es interpretado como datos para escribir en las direcciones de memoria sucesivas.

Si quiere conocer el valor actual de los registros del procesador, use:

```
show reg
```

O, puede mostrar un solo registro usando:

```
p $eax
```

y modificarlo haciendo:

```
set $eax new-value
```

Si usted quisiera llamar alguna función del kernel desde el DDB, solo debe decir:

```
call func(arg1, arg2, ...)
```

El valor devuelto será impreso en la pantalla.

Para un resumen de los procesos corriendo al estilo ps(1) use:

```
ps
```

Usted ya ha examinado la causa de que su kernel falle, y quiere reiniciar su equipo. Recuerde que, dependiendo de lo severo de las fallas que ocurrieron, algunas partes del kernel podrían no funcionar como se espera. Siga una de las siguientes acciones para apagar y reiniciar su equipo:

```
call diediedie()
```

Esto causará que su kernel haga un crash dump y reinicie, asi luego podrá analizar el dump a un nivel más alto con el kgdb. Este comando suele tener que acompanarse por otra instrucción `continue`. Para hacer esto hay un alias: `panic`.

```
call boot(0)
```

La cual podría ser una buena manera de apagar ordenadamente el sistema, hacer un `sync()` de todos los discos, y finalmente reiniciar. En tanto las interfaces de disco y de filesystem del kernel no esten danadas, esta podría ser una buena manera de hacer un apagado bastante prolijo.

```
call cpu_reset()
```

Es la ultima salida de los desastres y es practicamente lo mismo que presionar el Gran Boton Rojo.

Si usted necesita un breve sumario de los comandos, tipee:

```
help
```

De todos modos, es altamente recomendable tener una copia impresa de la página ddb(4) del manual a mano antes de la sesión de depuración. Recuerde que es bastante difícil leer el manual en línea mientras se está ejecutando el kernel paso a paso.

22.5. Depuración En-Línea Usando El GDB remoto

Esta característica ha sido soportada desde FreeBSD 2.2, y ya está en verdad muy bien pulida.

El GDB ha soportado *depuración remota* desde hace mucho tiempo. Esto se hace usando un protocolo muy simple a traves de una línea serie. A diferencia de los otros metodos descriptos anteriormente, hacen falta dos máquinas para hacer esto. Una va a proveer el entorno de depuración, incluyendo todos los archivos fuente, y una copia del ejecutable del kernel con todos los símbolos y la otra será la máquina a depurar que simplemente corre una copia de exactamente el mismo kernel (pero sin los símbolos de depuración).

Usted debería configurar el kernel en cuestión con `config -g`, incluir `DDB` en la configuración, y compilarla como siempre. Esto arrojará un ejecutable enorme, debido a la información de depuración. Copie este kernel a la máquina a depurar, quitele los simbolos con `strip -x`, e inicielo usando la opción `-d` en el prompt de inicio. Conecte el primer puerto serie de la máquina a cualquier puerto serie de la máquina que correrá el debugger. Ahora en la máquina que corre el debugger, vaya al directorio de compilación del kernel a depurar, y arranque el gdb:

```
% gdb -k kernel
GDB is free software and you are welcome to distribute copies of it
 under certain conditions; type "show copying" to see the conditions.
There is absolutely no warranty for GDB; type "show warranty" for details.
GDB 4.16 (i386-unknown-freebsd),
Copyright 1996 Free Software Foundation, Inc...
(kgdb)
```

Inicie la sesión de depuración remota haciendo (asumiendo que se usa el primer puerto serie):

```
(kgdb) target remote /dev/cuaa0
```

Ahora en la máquina a depurar (que entró al DDB justo antes de empezar a detectar los dispositivos), tipee:

```
Debugger("Boot flags requested debugger")
Stopped at Debugger+0x35: movb $0, edata+0x51bc
db> gdb
```

El DDB responderá diciendo:

```
Next trap will enter GDB remote protocol mode
```

Cada vez que tipee `gdb`, el modo se alternará entre el GDB remoto y el DDB local. Para forzar un siguiente trap inmediatamente, simplemente tipee `s` (avanza un paso). la máquina del debugger ahora ganará control sobre el kernel a depurar:

```
Remote debugging using /dev/cuaa0
Debugger (msg=0xf01b0383 "Boot flags requested debugger")
    at ../../i386/i386/db_interface.c:257
(kgdb)
```

Esta sesión puede usarse casi como cualquier otra sesión de GDB, incluyendo acceso completo al código fuente, ejecutarlo en modo-gud dentro de una ventana de Emacs (lo cual brinda la posibilidad de mostrar automáticamente el código fuente en otra ventana de Emacs) etc.

El GDB remoto tambien puede usarse para depurar LKMs. Primero compile el LKM con los simbolos de depuración:

```
# cd /usr/src/lkm/linux
# make clean; make COPTS=-g
```

Luego instale esta versión del módulo en la máquina a depurar, carguelo y use `modstat` para averiguar donde fue cargado:

```
# linux
# modstat
Type     Id Off Loadaddr Size Info     Rev Module Name
EXEC      0   4 f5109000 001c f510f010   1 linux_mod
```

Tome la dirección de carga (loadaddr) del módulo y sumele 0x20 (probablemente para contar el encabezado a.out). Esta es la dirección donde el código del módulo fue reubicado. Use el comando `add-symbol-file` en el GDB para informarle al debugger acerca del módulo:

```
(kgdb) add-symbol-file /usr/src/lkm/linux/linux_mod.o 0xf5109020
add symbol table from file "/usr/src/lkm/linux/linux_mod.o" at
text_addr = 0xf5109020? (y or n) y
(kgdb)
```

Ahora tiene acceso a todos los símbolos en el LKM.

22.6. Depurando Un Driver de Consola

Dado que el DDB necesita un driver de consola sobre el que correr, las cosas son mas complicadas si lo que falla es el propio driver de consola. Usted podría entonces recordar el uso de una consola en puerto serie (ya sea con un sector de inicio modificado, o especificando `-h` en el prompt `Boot:`) y colgar una terminal estándar en su primer puerto serie. El DDB funciona en cualquier driver de consola configurado, por supuesto tambien en una consola de puerto serie.

Capítulo 23. Emulacion Linux

Emulacion Linux

Capítulo 24. FreeBSD por Dentro

FreeBSD por Dentro

V. Apéndices

Capítulo 25. Obteniendo FreeBSD

Obteniendo FreeBSD

25.1. Servidores FTP

Capítulo 26. Bibliografía

Mientras que las páginas del manual proveen la referencia definitiva para partes individuales del sistema operativo FreeBSD, son notorias por no ilustrar como poner todas las piezas juntas para hacer que todo el sistema operativo funcione fácilmente. Debido a esto, no hay substituto para un buen libro de administración de sistemas UNIX y un buen manual del usuario.

26.1. Libros y revistas específicas sobre FreeBSD

Libros y revistas internacionales:

- Usando FreeBSD (http://freebsd.csie.nctu.edu.tw/~jdli/book.html) (en Chino).
- FreeBSD for PC 98'ers (en japonés), publicado por SHUWA System Co, LTD. ISBN 4-87966-468-5 C3055 P2900E.
- FreeBSD (en japonés), publicado por CUTT. ISBN 4-906391-22-2 C3055 P2400E.
- Introducción completa a FreeBSD (http://www.shoeisha.co.jp/pc/index/shinkan/97_05_06.htm) (en japonés), publicado por Shoeisha Co., Ltd (http://www.shoeisha.co.jp/). ISBN 4-88135-473-6 P3600E.
- Kit personal del principiante UNIX FreeBSD (http://www.ascii.co.jp/pb/book1/shinkan/detail/1322785.html) (en japonés), publicado por ASCII (http://www.ascii.co.jp/). ISBN 4-7561-1733-3 P3000E.
- Manual FreeBSD (traducción del japonés), publicado por ASCII (http://www.ascii.co.jp/). ISBN 4-7561-1580-2 P3800E.
- FreeBSD mit Methode (en alemán), publicado por Computer und Literatur Verlag/Vertrieb Hanser, 1998. ISBN 3-932311-31-0.
- Manual de instalación y utilización de FreeBSD (http://www.pc.mycom.co.uk/FreeBSD/install-manual.html) (en japonés), publicado por Mainichi Communications Inc. (http://www.pc.mycom.co.jp/).

Libros y revistas en inglés:

- The Complete FreeBSD (http://www.cdrom.com/titles/freebsd/bsdcomp_bkx.phtml), publicado por Walnut Creek CDROM (http://www.cdrom.com/).

26.2. Guías de usuario

- Computer Systems Research Group, UC Berkeley. *4.4BSD User's Reference Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-075-9
- Computer Systems Research Group, UC Berkeley. *4.4BSD User's Supplementary Documents*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-076-7
- *UNIX in a Nutshell*. O'Reilly & Associates, Inc., 1990. ISBN 093717520X
- Mui, Linda. *What You Need To Know When You Can't Find Your UNIX System Administrator*. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-104-6
- La Ohio State University (http://www-wks.acs.ohio-state.edu/) ha escrito un Curso de introducción a UNIX (http://www-wks.acs.ohio-state.edu/unix_course/unix.html) disponible en línea en formato HTML y postscript.
- Jpman Project, Japan FreeBSD Users Group (http://www.jp.FreeBSD.org/). FreeBSD User's Reference Manual (http://www.pc.mycom.co.jp/FreeBSD/urm.html) (traducción japonesa). Mainichi Communications Inc. (http://www.pc.mycom.co.jp/), 1998. ISBN4-8399-0088-4 P3800E.

26.3. Guías de administrador

- Albitz, Paul and Liu, Cricket. *DNS and BIND*, 2nd Ed. O'Reilly & Associates, Inc., 1997. ISBN 1-56592-236-0
- Computer Systems Research Group, UC Berkeley. *4.4BSD System Manager's Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-080-5
- Costales, Brian, et al. *Sendmail*, 2nd Ed. O'Reilly & Associates, Inc., 1997. ISBN 1-56592-222-0
- Frisch, Æleen. *Essential System Administration*, 2nd Ed. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-127-5
- Hunt, Craig. *TCP/IP Network Administration*. O'Reilly & Associates, Inc., 1992. ISBN 0-937175-82-X
- Nemeth, Evi. *UNIX System Administration Handbook*. 2nd Ed. Prentice Hall, 1995. ISBN 0131510517
- Stern, Hal *Managing NFS and NIS* O'Reilly & Associates, Inc., 1991. ISBN 0-937175-75-7
- Jpman Project, Japan FreeBSD Users Group (http://www.jp.FreeBSD.org/). FreeBSD System Administrator's Manual (http://www.pc.mycom.co.jp/FreeBSD/sam.html) (traducción japonesa). Mainichi Communications Inc. (http://www.pc.mycom.co.jp/), 1998. ISBN4-8399-0109-0 P3300E.

26.4. Guías de programadores

- Asente, Paul. *X Window System Toolkit*. Digital Press. ISBN 1-55558-051-3
- Computer Systems Research Group, UC Berkeley. *4.4BSD Programmer's Reference Manual*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-078-3
- Computer Systems Research Group, UC Berkeley. *4.4BSD Programmer's Supplementary Documents*. O'Reilly & Associates, Inc., 1994. ISBN 1-56592-079-1
- Harbison, Samuel P. and Steele, Guy L. Jr. *C: A Reference Manual*. 4rd ed. Prentice Hall, 1995. ISBN 0-13-326224-3
- Kernighan, Brian and Dennis M. Ritchie. *The C Programming Language.*. PTR Prentice Hall, 1988. ISBN 0-13-110362-9
- Lehey, Greg. *Porting UNIX Software*. O'Reilly & Associates, Inc., 1995. ISBN 1-56592-126-7
- Plauger, P. J. *The Standard C Library*. Prentice Hall, 1992. ISBN 0-13-131509-9
- Stevens, W. Richard. *Advanced Programming in the UNIX Environment*. Reading, Mass. : Addison-Wesley, 1992 ISBN 0-201-56317-7
- Stevens, W. Richard. *UNIX Network Programming*. 2nd Ed, PTR Prentice Hall, 1998. ISBN 0-13-490012-X
- Wells, Bill. "Writing Serial Drivers for UNIX". *Dr. Dobb's Journal*. 19(15), December 1994. pp68-71, 97-99.

26.5. El sistema operativo por dentro

- Andleigh, Prabhat K. *UNIX System Architecture*. Prentice-Hall, Inc., 1990. ISBN 0-13-949843-5
- Jolitz, William. "Porting UNIX to the 386". *Dr. Dobb's Journal*. January 1991-July 1992.
- Leffler, Samuel J., Marshall Kirk McKusick, Michael J Karels and John Quarterman *The Design and Implementation of the 4.3BSD UNIX Operating System*. Reading, Mass. : Addison-Wesley, 1989. ISBN 0-201-06196-1
- Leffler, Samuel J., Marshall Kirk McKusick, *The Design and Implementation of the 4.3BSD UNIX Operating System: Answer Book*. Reading, Mass. : Addison-Wesley, 1991. ISBN 0-201-54629-9
- McKusick, Marshall Kirk, Keith Bostic, Michael J Karels, and John Quarterman. *The Design and Implementation of the 4.4BSD Operating System*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-54979-4
- Stevens, W. Richard. *TCP/IP Illustrated, Volume 1: The Protocols*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-63346-9

- Schimmel, Curt. *Unix Systems for Modern Architectures*. Reading, Mass. : Addison-Wesley, 1994. ISBN 0-201-63338-8
- Stevens, W. Richard. *TCP/IP Illustrated, Volume 3: TCP for Transactions, HTTP, NNTP and the UNIX Domain Protocols*. Reading, Mass. : Addison-Wesley, 1996. ISBN 0-201-63495-3
- Vahalia, Uresh. *UNIX Internals – The New Frontiers*. Prentice Hall, 1996. ISBN 0-13-101908-2
- Wright, Gary R. and W. Richard Stevens. *TCP/IP Illustrated, Volume 2: The Implementation*. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-63354-X

26.6. Referencia de seguridad

- Cheswick, William R. and Steven M. Bellovin. *Firewalls and Internet Security: Repelling the Wily Hacker*. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-63357-4
- Garfinkel, Simson and Gene Spafford. *Practical UNIX Security*. 2nd Ed. O'Reilly & Associates, Inc., 1996. ISBN 1-56592-148-8
- Garfinkel, Simson. *PGP Pretty Good Privacy* O'Reilly & Associates, Inc., 1995. ISBN 1-56592-098-8

26.7. Referencia de hardware

- Anderson, Don and Tom Shanley. *Pentium Processor System Architecture*. 2nd Ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40992-5
- Ferraro, Richard F. *Programmer's Guide to the EGA, VGA, and Super VGA Cards*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-62490-7
- La corporación Intel publica documentación sobre sus CPUs, chipsets y estándares en su web para desarrolladores (http://developer.intel.com/), normalmente en archivos con formato PDF.
- Shanley, Tom. *80486 System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40994-1
- Shanley, Tom. *ISA System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40996-8
- Shanley, Tom. *PCI System Architecture*. 3rd ed. Reading, Mass. : Addison-Wesley, 1995. ISBN 0-201-40993-3
- Van Gilluwe, Frank. *The Undocumented PC*. Reading, Mass: Addison-Wesley Pub. Co., 1994. ISBN 0-201-62277-7

26.8. Historia de UNIX

- Lion, John *Lion's Commentary on UNIX, 6th Ed. With Source Code*. ITP Media Group, 1996. ISBN 1573980137
- Raymond, Eric S. *The New Hacker's Dictionary, 3rd edition*. MIT Press, 1996. ISBN 0-262-68092-0. Also known as the Jargon File (http://www.ccil.org/jargon/jargon.html)
- Salus, Peter H. *A quarter century of UNIX*. Addison-Wesley Publishing Company, Inc., 1994. ISBN 0-201-54777-5
- Simon Garfinkel, Daniel Weise, Steven Strassmann. *The UNIX-HATERS Handbook*. IDG Books Worldwide, Inc., 1994. ISBN 1-56884-203-1
- Don Libes, Sandy Ressler *Life with UNIX* — special edition. Prentice-Hall, Inc., 1989. ISBN 0-13-536657-7
- *The BSD family tree*. 1997. ftp://ftp.FreeBSD.org/pub/FreeBSD/FreeBSD-current/src/share/misc/bsd-family-tree o local (file:/usr/share/misc/bsd-family-tree) on a FreeBSD-current machine.
- *The BSD Release Announcements collection.* 1997. http://www.de.FreeBSD.org/de/ftp/releases/
- *Networked Computer Science Technical Reports Library*. http://www.ncstrl.org/
- *Antiguas releases BSD procedentes del Computer Systems Research Group (CSRG).* http://www.mckusick.com/csrg/: El paquete de 4 CDs cubre todas las versiones de BSD desde la 1BSD hasta la 4.4BSD y 4.4BSD-Lite2 (pero no la 2.11BSD, desafortunadamente). El último disco contiene el código fuente final y los archivos SCCS.

26.9. Diarios y revistas

- *The C/C++ Users Journal*. R&D Publications Inc. ISSN 1075-2838
- *Sys Admin — The Journal for UNIX System Administrators* Miller Freeman, Inc., ISSN 1061-2688

Capítulo 27. Recursos en Internet

Recursos en Internet

27.1. Listas de distribución

27.2. Grupos de News en USENET

Capítulo 28. Staff del Proyecto FreeBSD

El Proyecto FreeBSD está gestionado y operado por los siguientes grupos de personas:

28.1. El Core Team de FreeBSD

El core team de FreeBSD constituye el “Grupo de directores”, responsable de decidir el camino a seguir y mantener al proyecto en la dirección correcta, así cómo gestionar áreas específicas del proyecto.

(por órden alfabético de apellido):

- Satoshi Asami <asami@FreeBSD.org>
- Jonathan M. Bresler <jmb@FreeBSD.org>
- Andrey A. Chernov <ache@FreeBSD.org>
- Bruce Evans <bde@FreeBSD.org>
- Justin T. Gibbs <gibbs@FreeBSD.org>
- David Greenman <dg@FreeBSD.org>
- Jordan K. Hubbard <jkh@FreeBSD.org>
- Poul-Henning Kamp <phk@FreeBSD.org>
- Rich Murphey <rich@FreeBSD.org>
- Gary Palmer <gpalmer@FreeBSD.org>
- John Polstra <jdp@FreeBSD.org>
- Guido van Rooij <guido@FreeBSD.org>
- Søren Schmidt <sos@FreeBSD.org>
- Peter Wemm <peter@FreeBSD.org>
- Garrett Wollman <wollman@FreeBSD.org>
- Jörg Wunsch <joerg@FreeBSD.org>

28.2. Los desarrolladores de FreeBSD

Estas son las personas que tienen privilegios de committes y hacen el trabajo de ingeniería en el árbol de código fuente de FreeBSD. Todos los miembros del core team también son desarrolladores.

- Ugen J.S.Antsilevich <ugen@FreeBSD.org>
- Ade Barkah <mbarkah@FreeBSD.org>
- Stefan Bethke <stb@FreeBSD.org>
- Pierre Beyssac <pb@fasterix.freenix.org>
- Andrzej Bialecki <abial@FreeBSD.org>
- John Birrell <jb@cimlogic.com.au>
- Torsten Blum <torstenb@FreeBSD.org>
- Donald Burr <dburr@FreeBSD.org>
- Philippe Charnier <charnier@FreeBSD.org>
- Luoqi Chen <luoqi@FreeBSD.org>
- Eric J. Chet <ejc@FreeBSD.org>
- Kenjiro Cho <kjc@FreeBSD.org>
- Gary Clark II <gclarkii@FreeBSD.org>
- Archie Cobbs <archie@FreeBSD.org>
- Martin Cracauer <cracauer@FreeBSD.org>
- Adam David <adam@FreeBSD.org>
- Matthew Dillon <dillon@FreeBSD.org>
- Peter Dufault <dufault@FreeBSD.org>
- Frank Durda IV <uhclem@FreeBSD.org>
- Tor Egge <tegge@FreeBSD.org>
- Eivind Eklund <eivind@FreeBSD.org>
- Julian Elischer <julian@FreeBSD.org>
- Ralf S. Engelschall <rse@FreeBSD.org>
- Stefan Esser <se@FreeBSD.org>
- Sean Eric Fagan <sef@FreeBSD.org>
- Bill Fenner <fenner@FreeBSD.org>
- John Fieber <jfieber@FreeBSD.org>
- James FitzGibbon <jfitz@FreeBSD.org>
- Marc G. Fournier <scrappy@FreeBSD.org>

- Lars Fredriksen <lars@FreeBSD.org>
- Bill Fumerola <billf@FreeBSD.org>
- Thomas Gellekum <tg@FreeBSD.org>
- Brandon Gillespie <brandon@FreeBSD.org>
- Thomas Graichen <graichen@FreeBSD.org>
- Joe Greco <jgreco@FreeBSD.org>
- Rodney Grimes <rgrimes@FreeBSD.org>
- John-Mark Gurney <jmg@FreeBSD.org>
- Hiroyuki HANAI <hanai@FreeBSD.org>
- Peter Hawkins <thepish@FreeBSD.org>
- John Hay <jhay@FreeBSD.org>
- Wolfgang Helbig <helbig@FreeBSD.org>
- Guy Helmer <ghelmer@cs.iastate.edu>
- Eric L. Hernes <erich@FreeBSD.org>
- Nick Hibma <n_hibma@FreeBSD.org>
- Seiichirou Hiraoka <flathill@FreeBSD.org>
- Tatsumi Hosokawa <hosokawa@FreeBSD.org>
- Jeffrey Hsu <hsu@FreeBSD.org>
- Matthew Hunt <mph@FreeBSD.org>
- Jun-ichiro Itoh <itojun@itojun.org>
- Matthew Jacob <mjacob@FreeBSD.org>
- Gary Jennejohn <gj@FreeBSD.org>
- Nate Johnson <nsj@FreeBSD.org>
- L Jonas Olsson <ljo@FreeBSD.org>
- Takenori KATO <kato@FreeBSD.org>
- Andreas Klemm <andreas@FreeBSD.org>
- Motoyuki Konno <motoyuki@FreeBSD.org>
- Joseph Koshy <jkoshy@FreeBSD.org>
- Jun Kuriyama <kuriyama@FreeBSD.org>

- Greg Lehey <grog@FreeBSD.org>
- Jonathan Lemon <jlemon@FreeBSD.org>
- Don “Truck” Lewis <truckman@FreeBSD.org>
- Warner Losh <imp@FreeBSD.org>
- Scott Mace <smace@FreeBSD.org>
- Stephen McKay <mckay@FreeBSD.org>
- Kirk McKusick <mckusick@FreeBSD.org>
- Kenneth D. Merry <ken@FreeBSD.org>
- Ted Mittelstaedt <tedm@FreeBSD.org>
- Atsushi Murai <amurai@FreeBSD.org>
- Mark Murray <markm@FreeBSD.org>
- Masafumi NAKANE <max@FreeBSD.org>
- Alex Nash <alex@FreeBSD.org>
- Robert Nordier <rnordier@FreeBSD.org>
- David Nugent <davidn@blaze.net.au>
- David O’Brien <obrien@FreeBSD.org>
- Daniel O’Callaghan <danny@FreeBSD.org>
- L Jonas Olsson <ljo@FreeBSD.org>
- Steve Passe <fsmp@FreeBSD.org>
- Sujal Patel <smpatel@FreeBSD.org>
- Bill Paul <wpaul@FreeBSD.org>
- Joshua Peck Macdonald <jmacd@FreeBSD.org>
- Wes Peters <wes@FreeBSD.org>
- Steve Price <steve@FreeBSD.org>
- Mike Pritchard <mpp@FreeBSD.org>
- Doug Rabson <dfr@FreeBSD.org>
- James Raynard <jraynard@FreeBSD.org>
- Darren Reed <darrenr@FreeBSD.org>
- Geoff Rehmet <csgr@FreeBSD.org>

- Martin Renters <martin@FreeBSD.org>
- Paul Richards <paul@FreeBSD.org>
- Ollivier Robert <roberto@FreeBSD.org>
- Chuck Robey <chuckr@FreeBSD.org>
- Dima Ruban <dima@FreeBSD.org>
- Kenji SADA <sada@FreeBSD.org>
- Wolfram Schneider <wosch@FreeBSD.org>
- Andreas Schulz <ats@FreeBSD.org>
- Justin Seger <jseger@FreeBSD.org>
- Vanilla I. Shu <vanilla@FreeBSD.org>
- Michael Smith <msmith@FreeBSD.org>
- Dag-Erling C. Smørgrav <des@FreeBSD.org>
- Brian Somers <brian@FreeBSD.org>
- Gene Stark <stark@FreeBSD.org>
- Karl Strickland <karl@FreeBSD.org>
- Dmitrij Tejblum <dt@FreeBSD.org>
- Chris Timmons <cwt@FreeBSD.org>
- Paul Traina <pst@FreeBSD.org>
- Tim Vanderhoek <hoek@FreeBSD.org>
- Jacques Vidrine <nectar@FreeBSD.org>
- Steven Wallace <swallace@FreeBSD.org>
- Doug White <dwhite@FreeBSD.org>
- Nate Williams <nate@FreeBSD.org>
- Kazutaka YOKOTA <yokota@FreeBSD.org>
- Jean-Marc Zucconi <jmz@FreeBSD.org>
- Archie Cobbs <archie@FreeBSD.org>

28.3. El Proyecto de Documentación de FreeBSD

El Proyecto de Documentación (http://www.freebsd.org/docproj.html) es responsable de diferentes servicios, siendo ejecutado cada uno por un responsable y sus *ayudantes* (si los tiene):

Responsable del Proyecto de Documentación

Nik Clayton <nik@FreeBSD.org>

Webmaster

Wolfram Schneider <wosch@FreeBSD.org>

Editor del Handbook & FAQ

FAQ Maintainer <faq@FreeBSD.org>

Editor de Noticias

Nate Johnson <nsj@FreeBSD.org>

Ayudante: John Cavanaugh <john@FreeBSD.org>

Editor del FreeBSD Really-Quick NewsLetter

Chris Coleman <chrisc@vmunix.com>

Editor de la Galería y sección Comercial

Nate Johnson <nsj@FreeBSD.org>

Ayudante: Charles A. Wimmer <cawimm@FreeBSD.org>

Style Police & Art Director

Chris Watson <opsys@open-systems.net>

Ingeniero de las Bases de Datos

Mark Mayo <mark@vmunix.com>

Ingeniero de CGI

Stefan Bethke <stb@FreeBSD.org>

Bottle Washing

Nate Johnson <nsj@FreeBSD.org>

COnversión de LinuxDoc a DocBook

Nik Clayton <nik@FreeBSD.org>

28.4. Quién es responsable de qué

Arquitecto Principal

David Greenman <dg@FreeBSD.org>

Responsable del Proyecto de Documentación (http://www.freebsd.org/docproj/docproj.html)

Nik Clayton <nik@FreeBSD.org>

Internationalización

Andrey A. Chernov <ache@FreeBSD.org>

Networking

Garrett Wollman <wollman@FreeBSD.org>

Postmaster

Jonathan M. Bresler <jmb@FreeBSD.org>

Coordinador de Release

Jordan K. Hubbard <jkh@FreeBSD.org>

Relaciones Públicas & Corporativas

Jordan K. Hubbard <jkh@FreeBSD.org>

Responsable de Seguridad (http://www.freebsd.org/security/)

Guido van Rooij <guido@FreeBSD.org>

>Responsables del CVS (http://www.freebsd.org/support.html#cvs)

Principal: Peter Wemm <peter@FreeBSD.org>

Asistente: John Polstra <jdp@FreeBSD.org>

Internacional (Crypto): Mark Murray <markm@FreeBSD.org>

Responsable de Ports (http://www.freebsd.org/ports/)

Satoshi Asami <asami@FreeBSD.org>

XFree86 Project, Inc.

Rich Murphey <rich@FreeBSD.org>

Soporte Usenet

Jörg Wunsch <joerg@FreeBSD.org>

Administrador GNATS (http://www.freebsd.org/support.html#gnats)

Steve Price <steve@FreeBSD.org>

Webmaster (http://www.freebsd.org/internal/)

Wolfram Schneider <wosch@FreeBSD.org>

Capítulo 29. PGP keys

En caso de que necesites verificar una firma o enviar un mail encriptado a alguno de los responsables o miembros del core team, aquí tienes una serie de claves para uso.

29.1. Responsables

29.1.1. Responsable de Seguridad de FreeBSD <security-officer@freebsd.org>

```
FreeBSD Security Officer <security-officer@freebsd.org>
Fingerprint = 41 08 4E BB DB 41 60 71  F9 E5 0E 98 73 AF 3F 11

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.3i

mQCNAzF7MY4AAAEEAK7qBgPuBejER5HQbQlsOldk3ZVWXlRj54raz3IbuAUrDrQL
h3g57T9QY++f3Mot2LAf5lDJbsMfWrtwPrPwCCFRYQd6XH778a+l4ju5axyjrt/L
Ciw9RrOC+WaPv3lIdLuqYge2QRC1LvKACIPNbIcgbnLeRGLovFUuHi5z0oilAAUR
tDdGcmVlQlNEIFNlY3VyaXR5IE9mZmljZXIgPHNlY3VyaXR5LW9mZmljZXJAZnJl
ZWJzZC5vcmc+iQCVAwUQMX6yrOJgpPLZnQjrAQHyowQA1Nv2AY8vJIrdp2ttV6RU
tZBYnI7gTO3sFC2bhIHsCvfVU3JphfqWQ7AnTXcD2yPjGcchUfc/EcL1tSlqW4y7
PMP4GHZp9vHog1NAsgLC9Y1P/1cOeuhZ0pDpZZ5zxTo6TQcCBjQA6KhiBFP4TJql
3olFfPBh3B/Tu3dqmEbSWpuJAJUDBRAxez3C9RVb+45ULV0BAak8A/9JIG/jRJaz
QbKom6wMw852C/Z0qBLJy7KdN30099zMjQYeC9PnlkZ0USjQ4TSpC8UerYv6IfhV
nNY6gyF2Hx4CbEFlopnfA1c4yxtXKti1kSN6wBy/ki3SmqtfDhPQ4Q31p63cSe5A
3aoHcjvWuqPLpW4ba2uHVKGP3g7SSt6AOYkAlQMFEDF8mz0ff6kIA1j8vQEBmZcD
/REaUPDRx6qr1XRQlMs6pfgNKEwnKmcUzQLCvKBnYYGmD5ydPLxCPSFnPcPthaUb
5zVgMTjfjS2fkEiRrua4duGRgqN4xY7VRAsIQeMSITBOZeBZZf2oa9Ntidr5PumS
9uQ9bvdfWMpsemk2MaRG9BSoy5Wvy8VxROYYUwpT8Cf2iQCVAwUQMXsyqWtaZ42B
sqd5AQHKjAQAvolI30Nyu3IyTfNeCb/DvOe9tlOn/o+VUDNJiE/PuBe1s2Y94a/P
BfcohpKC2kza3NiW6lLTp00OWQsuu0QAPc02vYOyseZWy4y3Phnw60pWzLcFdemT
0GiYS5Xm1o9nAhPFciybn9j1q8UadIlIq0wbqWgdInBT8YI/l4f5sf6JAJUDBRAx
ezKXVS4eLnPSiKUBAc5OBACIXTlKqQC3B53qt7bNMV46m81fuw1PhKaJEI033mCD
ovzyEFFQeOyRXeu25Jg9Bq0Sn37ynISucHSmt2tUD5W0+p1MUGyTqnfqejMUWBzO
v4Xhp6a8RtDdUMBOTtro16iulGiRrCKxzVgEl4i+9Z0ZiE6BWlg5AetoF5n3mGk1
lw==
=ipyA
---END PGP PUBLIC KEY BLOCK---
```

29.1.2. Warner Losh <imp@FreeBSD.org>

```
Warner Losh <imp@village.org>
 aka <imp@freebsd.org>
Fingerprint = D4 31 FD B9 F7 90 17 E8 37 C5 E7 7F CF A6 C1 B9
---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.2

mQCNAzDzTiAAAAEEAK8D7KWEbVFUrmlqhUEnAvphNIqHEbqqT8s+c5f5c2uHtlcH
V4mV2TlUaDSVBN4+/D70oHmZc4IgiQwMPCWRrSezg9z/MaKlWhaslc8YT6Xc1q+o
EP/fAdKUrq49H0QQbkQk6Ks5wKW6v9AOvdmsS6ZJEcet6d9G4dxynu/2qPVhAAUR
tCBNLiBXYXJuZXIgTG9zaCA8aW1wQHZpbGxhZ2Uub3JnPokAlQMFEDM/SK1VLh4u
c9KIpQEBFPsD/1n0YuuUPvD4CismZ9bx9M84y5sxLolgFEfP9Ux196ZSeaPpkA0g
C9YX/IyIy5VHh3372SDWN5iVSDYPwtCmZziwIV2YxzPtZw0nUu82P/Fn8ynlCSWB
5povLZmgrWijTJdnUWI0ApVBUTQoiW5MyrNN51H3HLWXGoXMgQFZXKWYiQCVAwUQ
MzmhkfUVW/uOVC1dAQG3+AP/T1HL/5EYF0ij0yQmNTzt1cLt0b1e3N3zN/wPFFWs
BfrQ+nsv1zw7cEgxLtktk73wBGM9jUIdJu8phgLtl5a0m9UjBq5oxrJaNJr6UTxN
a+sFkapTLT1g84UFUO/+8qRB12v+hZr2WeXMYjHAFUT18mp3xwjW9DUV+2fW1Wag
YDKJAJUDBRAzOYK1s1pi61mfMj0BARBbA/930CHswOF0HIr+4YYUs1ejDnZ2J3zn
icTZhl9uAfEQq++Xor1x476j67Z9fESxyHltUxCmwxsJ1uOJRwzjyEoMlyFrIN4C
dE0C8g8BF+sRTt7VLURLERvlBvFrVZueXSnXvmMoWFnqpSpt3EmN6TNaLe8Cm87a
k6EvQy0dpnkPKokAlQMFEDD9Lorccp7v9qj1YQEBrRUD/3N4cCMWjzsIFp2Vh9y+
RzUrblyF84tJyA7Rr1p+A7dxf7je3Zx5QMEXosWL1WGnS5vC9YH2WZwv6sCU61gU
rSy9z8KHlBEHh+Z6fdRMrjd9byPf+n3cktT0NhS23oXB1ZhNZcB2KKhVPlNctMqO
3gTYx+Nlo6xqjR+J2NnBYU8p =7fQV
---END PGP PUBLIC KEY BLOCK---
```

29.2. Miembros del Core Team

29.2.1. Satoshi Asami <asami@FreeBSD.org>

```
Satoshi Asami <asami@cs.berkeley.edu>
          aka <asami@FreeBSD.ORG>
Fingerprint = EB 3C 68 9E FB 6C EB 3F  DB 2E 0F 10 8F CE 79 CA

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.2

mQCNAzPVyoQAAAEEAL7W+kipxB171Z4SVyyL9skaA7hG3eRsSOWk7lfvfUBLtPog
f3OKwrApoc/jwLf4+Qpdzv5DLEt/6Hd/clskhJ+q1gMNHyZ5ABmUxrTRRNvJMTrb
```

```
3fPU3oZj7sL/MyiFaT1zF8EaMP/iS2ZtcFsbYOqGeA8E/58uk4NA0SoeCNiJAAUR
tCVTYXRvc2hpIEFzYW1pIDxhc2FtaUBjcy5iZXJrZWxleS5lZHU+iQCVAwUQM/AT
+EqGN2HYnOMZAQF11QP/eSXb2FuTb1yX5yoo1Im8YnIk1SEgCGbyEbOMMBznVNDy
5g2TAD0ofLxPxy5Vodjg8rf+lfMVtO5amUH6aNcORXRncE83T10JmeM6JEp0T6jw
zOHKz8jRzygYLBayGsNIJ4BGxa4LeaGxJpO1ZEvRlNkPH/YEXK5oQmq9/DlrtYOJ
AEUDBRAz42JT8ng6GBbVvu0BAU8nAYCsJ8PiJpRUGlrz6rxjX8hqM1v3vqFHLcG+
G52nVMBSy+RZBgzsYIPwI5EZtWAKb22JAJUDBRAz4QBWdbtuOHaj97EBAaQPA/46
+NLUp+Wubl90JoonoXocwAg88tvAUVSzsxPXj0lvypAiSI2AJKsmn+5PuQ+/IoQy
lywRsxiQ5GD7C72SZ1yw2WI9DWFeAi+qa4b8n9fcLYrnHpyCY+zxEpu4pam8FJ7H
JocEUZz5HRoKKOLHErzXDiuTkkm72b1glmCqAQvnB4kAlQMFEDPZ3gyDQNEqHgjY
iQEBFfUEALu2C0uo+1Z7C5+xshWRYY5xNCzK20O6bANVJ+CO2fih96KhwsMof3lw
fDso5HJSwgFd8WT/sR+Wwzz6BAE5UtgsQq5GcsdYQuGI1yIlCYUpDp5sgswNm+OA
bX5a+r4F/ZJqrqT1J56Mer0VVsNfe5nIRsjd/rnFAFVfjcQtaQmjiQCVAwUQM9uV
mcdm8Q+/vPRJAQELHgP9GqNiMpLQlZig17fDnCJ73P0e5t/hRLFehZDlmEI2TK7j
Yeqbw078nZgyyuljZ7YsbstRIsWVCxobX5eH1kX+hIxuUqCAkCsWUY4abG89kHJr
XGQn6X1CX7xbZ+b6b9jLK+bJKFcLSfyqR3M2eCyscSiZYkWKQ5l3FYvbUzkeb6K0
IVNhdG9zaGkgQXNhbWkgPGFzYW1pQEZyZWVCU0QuT1JHPg==
=39SC
---END PGP PUBLIC KEY BLOCK---
```

29.2.2. Jonathan M. Bresler <jmb@FreeBSD.org>

```
Jonathan M. Bresler <jmb@FreeBSD.org>
f16    Fingerprint16 = 31 57 41 56 06 C1 40 13  C5 1C E3 E5 DC 62 0E FB

---BEGIN PGP PUBLIC KEY BLOCK---
Version: PGPfreeware 5.0i for non-commercial use

mQCNAzG2GToAAAEEANI6+4SJAAgBpl53XcfEr1M9wZyBqC0tzpie7Zm4vhv3hO8s
o5BizSbcJheQimQiZAY4OnlrCpPxijMFSaihshs/VMAz1qbisUYAMqwGEO/T4QIB
nWNo0Q/qOniLMxUrxS1RpeW5vbghErHBKUX9GVhxbiVfbwc4wAHbXdKX5jjdAAUR
tCVKb25hdGhhbiBNLiBCcmVzbGVyIDxqbWJARnJlZUJTRC5PUkc+iQCVAwUQNbtI
gAHbXdKX5jjdAQHamQP+OQr10QRknamIPmuHmFYJZ0jU9XPIvTTMuOiUYLcXlTdn
GyTUuzhbEywgtOldW2V5iA8platXThtqC68NsnN/xQfHA5xmFXVbayNKn8H5stDY
2s/4+CZ06mmJfqYmONF1RCbUk/M84rVT3Gn2tydsxFh4Pm32lf4WREZWRiLqmw+J
AJUDBRA0DfF99RVb+45ULV0BAcZ0BACCydiSUG1VR0a5DBcHdtin2iZMPsJUPRqJ
tWvP6VeI8OFpNWQ4LW6ETAvn35HxV2kCcQMyht1kMD+KEJz7r8Vb94TS7KtZnNvk
2D1XUx8Locj6xel5c/Lnzlnnp7Bp1XbJj2u/NzCaZQ0eYBdP/k7RLYBYHQQln5x7
BOuiRJNVU4kAlQMFEDQLcShVLh4uc9KIpQEBJv4D/3mDrD0MM9EYOVuyXik3UGVI
8quYNA9ErVcLdt10NjYc16VI2HOnYVgPRag3Wt7W8wlXShpokfC/vCNt7f5JgRf8
h2a1/MjQxtlD+4/Js8k7GLa53oLon6YQYk32IEKexoLPwIRO4L2BHWa3GzHJJSP2
aTR/Ep90/pLdAOu/oJDUiQCVAwUQMqyL0LNaYutZnzI9AQF25QP9GFXhBrz2tiWz
2+0gWbpcGNnyZbfsVjF6ojGDdmsjJMyWCGw49XR/vPKYIJY9EYo4t49GIajRkISQ
```

```
NNiIz22fBAjT2uY9YlvnTJ9NJleMfHr4dybo7oEKYMWWijQzGjqf2m8wf9OaaofE
KwBX6nxcRbKsxm/BVLKczGYl3XtjkcuJAJUDBRA1ol5TZWCprDT5+dUBATzXA/9h
/ZUuhoRKTWViaistGJfWi26FB/Km5nDQBr/Erw3XksQCMwTLyEugg6dahQ1u9Y5E
5tKPxbB69eF+7JXVHE/z3zizR6VL3sdRx74TPacPsdhZRjChEQc0htLLYAPkJrFP
VAzAlSlm7qd+MXf8fJovQs6xPtZJXukQukPNlhqZ94kAPwMFEDSH/kF4tXKgazlt
bxECfk4AoO+VaFVfguUkWX10pPSSfvPyPKqiAJ4xn8RSIe1ttmnqkkDMhLh00mKj
lLQuSm9uYXRoYW4gTS4gQnJlc2xlciA8Sm9uYXRoYW4uQnJlc2xlckBVU2kubmV0
PokAlQMFEDXbdSkB213Sl+Y43QEBV/4D/RLJNTrtAqJ1ATxXWv9g8Cr3/YF0GTmx
5dIrJOpBup7eSSmiM/BL9Is4YMsoVbXCI/8TqA67TMICvq35PZU4wboQB8DqBAr+
gQ8578M7Ekw1OAF6JXY6AF2P8k7hMcVBcVOACELPT/NyPNByG5QRDoNmlsokJaWU
/2ls4QSBZZlb
=zbCw
---END PGP PUBLIC KEY BLOCK---
```

29.2.3. Andrey A. Chernov <ache@FreeBSD.org>

```
Andrey A. Chernov <ache@FreeBSD.org>
       aka <ache@nagual.pp.ru>
Key fingerprint = 33 03 9F 48 33 7B 4A 15  63 48 88 0A C4 97 FD 49

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.3ia

mQCNAiqUMGQAAAEEAPGhcD6A2Buey5LYz0sphDLpVgOZc/bb9UHAbaGKUAGXmafs
Dcb2HnsuYGgX/zrQXuCi/wIGtXcZWB97APtKOhFsZnPinDR5n/dde/mw9FnuhwqD
m+rKSL1HlN0z/Msa5y7g16760wHhSR6NoBSEG5wQAHIMMq7Q0uJgpPLZnQjrAAUT
tCVBbmRyZXkgQS4gQ2hlcm5vdiA8YWNoZUBuYWd1YWwucHAucnU+iQCVAwUQM2Ez
u+JgpPLZnQjrAQEyugP8DPnS8ixJ5OeuYgPFQf5sy6l+LrB6hyaS+lgsUPahWjNY
cnaDmfda/q/BV5d4+y5rlQe/pjnYG7/yQuAR3jhlXz8XDrqlBOnW9AtYjDt5rMfJ
aGFTGXAPGZ6k6zQZE0/YurT8ia3qjvuZm3Fw4NJrHRx7ETHRvVJDvxA6Ggsvmr20
JEFuZHJleSBBLiBDaGVybm92IDxhY2hlQEZyZWVCU0Qub3JnPokAlQMFEDR5uVbi
YKTy2Z0I6wEBLgED/2mn+hw4/3peLx0Sb9LNx//NfCCkVefSf2G9Qwhx6dvwbX7h
mFca97h7BQN4GubU1Z5Ffs6TeamSBrotBYGmOCwvJ6S9WigF9YHQIQ3B4LEjskAt
pcjU583y42zM11kkvEuQU2Gde61daIylJyOxsgpjSWpkxq50fgY2kLMfgl/ftCZB
bmRyZXkgQS4gQ2hlcm5vdiA8YWNoZUBuaWV0enNjaGUubmV0PokAlQMFEDR5svDi
YKTy2Z0I6wEBOTQD/0OTCAXIjuak363mjERvzSkVsNtIH9hA1l0w6Z95+iH0fHrW
xXKT0vBZE0y0Em+S3cotLL0bMmVE3F3D3GyxhBVmgzjyx0NYNoiQjYdi+6g/PV30
Cn4vOO6hBBpSyI6vY6qGNqcsawuRtHNvK/53MpOfKwSlICEBYQimcZhkci+EtCJB
bmRyZXkgQS4gQ2hlcm5vdiA8YWNoZUBuYWd1YWwucnU+iQCVAwUQMcm5HeJgpPLZ
nQjrAQHwvQP9GdmAf1gdcuayHEgNkc11macPH11cwWjYjzA2YoecFMGV7iqKK8QY
rr1MjbGXf8DAG8Ubfm0QbI8Lj8iG3NgqIru0c72UuHGSn/APfGGG0AtPX5UK/k7B
gI0Ca2po6NA5nrSp8tDsdEz/4gyea84RXl2prtTf5Jj07hflbRstGXK0MkFuZHJl
eSBBLiBDaGVybm92LCBCbGFjayBNYWdlIDxhY2hlQGFzdHJhbC5tc2suc3U+iQCV
```

```
AwUQMCsAo5/rGryoL8h3AQHq1QQAidyNFqA9hvrmMcjpY7csJVFlGvj574Wj4GPa
o3pZeuQaMBmsWqaXLYnWU/Aldb6kTz6+nRcQX50zFH0THSPfApwEW7yybSTI5apJ
mWT3qhKN2vmLNg2yNzhqLTzHLD1lH3i1pfQq8WevrNfjLUco5S/VuekTma/osnzC
Cw7fQzCJAJUDBRAwKvwoa1pnjYGyp3kBARihBACoXr3qfG65hFCyKJISmjOvaoGr
anxUIkeDS0yQdTHzhQ+dwB1OhhK15E0Nwr0MKajLMm90n6+Zdb5y/FIjpPriu8dI
rlHrWZlewa88eEDM+Q/NxT1iYg+HaKDAE171jmLpSpCL0MiJtO0i36L3ekVD7Hv8
vffOZHPSHirIzJOZTYkAlQMFEDAau6zFLUdtDb+QbQEBQX8D/AxwkYeFaYxZYMFO
DHIvSk23hAsjCmUA2Uil1FeWAusb+o8xRfPDc7TnosrIifJqbF5+fcHCG5VSTGlh
Bhd18YWUeabf/h9O2BsQX55yWRuB2x3diJ1xI/VVdG+rxlMCmE4ZR1Tl9x+Mtun9
KqKVpB39VlkCBYQ3hlgNt/TJUY4riQCVAwUQMBHMmyJRltlmbQBRAQFQkwP/YC3a
hs3ZMMoriOlt3ZxGNUUPTF7rIER3j+c7mqGG46dEnDB5sUrkzacpoLX5sj1tGR3b
vz9a4vmk1Av3KFNNvrZZ3/BZFGpq3mCTiAC9zsyNYQ8L0AfGIUO5goCIjqwOTNQI
AOpNsJ5S+nMAkQB4YmmNlI6GTb3D18zfhPZ6uciJAJUCBRAwD0sl4uW74fteFRkB
AWsAA/9NYqBRBKbmltQDpyK4+jBAYjkXBJmARFXKJYTlnTgOHMpZqoVyW96xnaa5
MzxEiu7ZWm5oL10QDIp1krkBP2KcmvfSMMHb5aGCCQc2/P8NlfXAuHtNGzYiI0UA
Iwi8ih/S1liVfvnqF9uV3d3koE7VsQ9OA4Qo0ZL2ggW+/gEaYIkAlQMFEDAOz6qx
/IyHe3rl4QEBIvYD/jIr8Xqo/2I5gncghSeFR01n0vELFIvaF4cHofGzyzBpYsfA
+6pgFI1IM+LUF3kbUkAY/2uSf9U5ECcaMCTWCwVgJVO+oG075SHEM4buhrzutZiM
1dTyTaepaPpTyRMUUx9ZMMYJs7sbqLId1eDwrJxUPhrBNvf/w2W2sYHSY8cdiQCV
AwUQMAzqgHcdkq6JcsfBAQGTxwQAtgeLFi2rhSOdllpDXUwz+SS6bEjFTWgRsWFM
y9QnOcqryw7LyuFmWein4jasjY033JsODfWQPiPVNA3UEnXVg9+n8AvNMPO8JkRv
Cn1eNg0VaJy9J368uArio93agd2Yf/R5r+QEuPjIssVk8hdcy/luEhSiXWf6bLMV
HEA0J+OJAJUDBRAwDUi+4mCk8tmdCOsBAatBBACHB+qtW880seRCDZLjl/bT1b14
5po60U7u6a3PEBkY0NA72tWDQuRPF/Cn/0+VdFNxQUsgkrbwaJWOoi0KQsvlOm3R
rsxKbn9uvEKLxExyKH3pxp76kvz/lEWwEeKvBK+84Pb1lzpG3W7u2XDfi3VQPTi3
5SZMAHc6C0Ct/mjNlYkAlQMFEDAMrPD7wj+NsTMUOQEBJckD/ik4WsZzm2qOx9Fw
erGq7Zwchc+Jq1YeN5PxpzqSf4AG7+7dFIn+oe6X2FcIzgbYY+IfmgJIHEVjDHH5
+uAXyb6l4iKc89eQawO3t88pfHLJWbTzmnvgz2cMrxt94HRvgkHfvcpGEgbyldq6
EB33OunazFcfZFRIcXk1sfyLDvYE
=1ahV
---END PGP PUBLIC KEY BLOCK---
```

29.2.4. Jordan K. Hubbard <jkh@FreeBSD.org>

```
Jordan K. Hubbard <jkh@FreeBSD.org>
Fingerprint = 3C F2 27 7E 4A 6C 09 0A  4B C9 47 CD 4F 4D 0B 20

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.3ia

mQCNAzFjX0IAAAEEAML+nm9/kDNPp43ZUZGjYkm2QLtoC1Wxr8JulZXqk7qmhYcQ
jvX+fyoriJ6/7ZlnLe2oG5j9tZOnRLPvMaz0g9CpW6Dz3nkXrNPkmOFV9B8D94Mk
tyFeRJFqnkCuqBj6D+H8FtBwEeeTecSh2tJ0bZZTXnAMhxeOdvUVW/uOVC1dAAUR
```

```
tCNKb3JkYW4gSy4gSHViYmFyZCA8amtoQEZyZWVCU0Qub3JnPokBFQMFEDXCTXQM
j46yp4IfPQEBwO8IAIN0J09AXBf86dFUTFGcAMrEQqOF5IL+KGorAjzuYxERhKfD
ZV7jA+sCQqxkWfcVcE20kVyVYqzZIkio9a5zXP6TwA247JkPt54S1PmMDYHNlRIY
laXlNoji+4q3HP2DfHqXRT2859rYpm/fG/v6pWkos5voPKcZ2OFEp9W+Ap88oqw+
5rx4VetZNJq1Epmis4INj6XqNqj85+MOOIYE+f445ohDM6B/Mxazd6cHFGGIR+az
VjZ6lCDMLjzhB5+FqfrDLYuMjqkMTR5z9DL+psUvPlCkYbQ11NEWtEmiIWjUcNJN
GCxGzv5bXk0XPu3ADwbPkFE2usW1cSM7AQFiwuyJAJUDBRAxe+Q9a1pnjYGyp3kB
AV7XA/oCSL/Cc2USpQ2ckwkGpyvIkYBPszIcabSNJAzm2hsU9Qa6WOPxD8olDddB
uJNiW/gznPC4NsQ0N8Zr4IqRX/TTDVf04WhLmd8AN9SOrVv2q0BKgU6fLuk979tJ
utrewH6PR2qBOjAaR0FJNk4pcYAHeT+e7KaKy96YFvWKIyDvc4kAlQMFEDF8ldof
f6kIA1j8vQEBDH4D/0Zm0oNlpXrAE1EOFrmp43HURHbij8n0Gra1w9sbfo4PV+/H
U8ojTdWLy6r0+prH7NODCkgtIQNpqLuqM8PF2pPtUJj9HwTmSqfaT/LMztfPA6PQ
csyT7xxdXl0+4xTDl1avGSJfYsI8XCAy85cTs+PQwuyzugE/iykJO1Bnj/paiQCV
AwUQMXvlBvUVW/uOVC1dAQF2fQP/RfYC6RrpFTZHjo2qsUHSRk0vmsYfwG5NHP5y
oQBMsaQJeSckN4n2JOgR4T75U4vS62aFxgPLJP3lOHkU2Vc7xhAuBvsbGr5RP8c5
LvPOeUEyz6ZArp1KUHrtcM2iK1FBOmY4dOYphWyWMkDgYExabqlrAq7FKZftpq/C
BiMRuaw=
=C/Jw
---END PGP PUBLIC KEY BLOCK---
```

29.2.5. Poul-Henning Kamp <phk@FreeBSD.org>

```
Poul-Henning Kamp <phk@FreeBSD.org>
Fingerprint = A3 F3 88 28 2F 9B 99 A2  49 F4 E2 FA 5A 78 8B 3E

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.3ia

mQCNAzAdpMIAAAEEALHDgrFUwhZtb7PbXg3upELoDVEUPFRwnmpJH1rRqyROUGcI
ooVe7u+FQlIs5OsXK8ECs/5Wpe2UrZSzHvjwBYOND5H42YtI5UULZLRCo5bFfTVA
K9Rpo5icfTsYihrzU2nmnycwFMk+jYXyT/ZDYWDP/BM9iLjj0x9/qQgDWPy9AAUR
tCNQb3VsLUhlbm5pbmcgS2FtcCA8cGhrQEZyZWVCU0Qub3JnPokAlQMFEDQQ0aZ1
u244dqP3sQEBu4ID/jXFFeJgs2MdTDNOZM/FbfDhI4qxAbYUsqS3+Ra16yd8Wd/A
jV+IHJE2NomFWl8UrUjCGinXiwzPgK1OfFJrS9Og1wQLvAl0X84BA8MTP9BQr4w7
6I/RbksgUSrVCIO8MJwlydjSPocWGBeXlVjbZxXzyuJk7H+TG+zuI5BuBcNIiQCV
AwUQMwYr2rNaYutZnzI9AQHiIQP/XxtBWFXaBRgVLEhRNpS07YdU+LsZGlLOZehN
9L4UnJFHQQPNOpMey2gF7Y95aBOw5/1xS5vlQpwmRFCntWsm/gqdzK6rulfr1r5A
y94LO5TAC6ucNu396Y4vo1TyD1STnRC466KlvmtQtAtFGgXlORWLL9URLzcRFd1h
D0yXd9aJAJUDBRAxfo19a1pnjYGyp3kBAQqyA/4v64vP3l1F0Sadn6ias761hkz/
SMdTuLzILmofSCC4o4KWMjiWJHs2Soo41QlZi1+xMHzV32JKiwFlGtPHqL+EHyXy
Q4H3vmf9/1KF+0XCaMtgI0wWUMziPSTJK8xXbRRmMDK/0F4TnVVaUhnmf+h5K7O6
XdmejDTa0X/NWcicmIkAlQMFEDF8lef1FVv7jlQtXQEBcnwD/0ro1PpUtlkLmreD
tsGTkNa7MFLegrYRvDDrHOwPZH152W2jPUncY+eArQJakeHiTDmJNpFagLZglhE0
```

```
bqJyca+UwCXX+6upAclWHEBMg2byiWMMqyPVEEnpUoHM1sIkgdNWlfQAmipRBfYh
2LyCgWvR8CbtwPYIFvUmGgB3MR87iQCVAwUQMUseXB9/qQgDWPy9AQGPkwP/WEDy
El2Gkvua9COtMAifot2vTwuvWWpNopIEx0Ivey4aVbRLD90gGCJw8OGDEtqFPcNV
8aIiy3fYVKXGZZjvCKd7zRfhNmQn0eLDcymq2OX3aPrMc2rRlkT4Jx425ukR1gsO
qiQAgw91aWhY8dlw/EKzk8ojm52x4VgXaBACMjaJAJUDBRAxOUOg72G56RHVjtUB
AbL4A/9HOn5Qa0lq9tKI/HkSdc5fGQD/66VdCBAb292RbB7CS/EM07MdbcqRRYIa
0+0gwQ3OdsWPdCVgH5RIhp/WiC+UPkR1cY8N9Mg2kTwJfZZfNqN+BgWlgRMPN27C
OhYNl8Q33Nl9CpBLrZWABF44jPeT0EvvTzP/5ZQ7T75EsYKYiYkAlQMFEDDmryQA
8tkJ67sbQQEBPdsEALCj6v1OBuJLLJTlxmmrkqAZPVzt5QdeO3Eqa2tcPWcU0nqP
vHYMzZcZ7oFg58NZsWrhSQQDIB5e+K65Q/h6dC7W/aDskZd64jxtEznX2kt0/MOr
8OdsDis1K2f9KQftrAx81KmVwW4Tqtzl7NWTDXt44fMOtibCwVq8v2DFkTJy
=JKbP
---END PGP PUBLIC KEY BLOCK---
```

29.2.6. Rich Murphey <rich@FreeBSD.org>

```
Rich Murphey <rich@FreeBSD.org>
fingerprint = AF A0 60 C4 84 D6 0C 73  D1 EF C0 E9 9D 21 DB E4

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.2

mQCNAy97V+MAAAEEALiNM3FCwm3qrCe81E20UOSlNclOWfZHNAyOyj1ahHeINvo1
FBF2Gd5Lbj0y8SLMno5yJ6P4F4r+x3jwHZrzAIwMs/lxDXRtB0VeVWnlj6a3Rezs
wbfaTeSVyh5JohEcKdoYiMG5wjATOwK/NAwIPthB1RzRjnEeer3HI3ZYNEOpAAUR
tCRSaWNoIE11cnBoZXkgPHJpY2hAbGFtcHJleS51dG1iLmVkdT6JAJUDBRAve15W
vccjdlg0Q6kBAZTZBACcNd/LiVnMFURPrO4pVRn1sVQeokVX7izeWQ7siE31Iy7g
Sb97WRLEYDi686osaGfsuKNA87Rm+q5F+jxeUV4w4szoqp60gGvCbD0KCB2hWraP
/2s2qdVAxhfcoTin/Qp1ZWvXxFF7imGA/IjYIfB42VkaRYu6BwLEm3YAGfGcSw==
=QoiM
---END PGP PUBLIC KEY BLOCK---
```

29.2.7. John Polstra <jdp@FreeBSD.org>

```
John D. Polstra <jdp@polstra.com>
Fingerprint = 54 3A 90 59 6B A4 9D 61  BF 1D 03 09 35 8D F6 0D

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.2

mQCNAzMElMEAAAEEALizp6ZW9QifQgWoFmG3cXhzQ1+Gt+a4S1adC/TdHdBvw1M/
```

```
I6Ok7TC0dKF8blW3VRgeHo4F3XhGn+n9MqIdboh4HJC5Iiy63m98sVLJSwyGO4oM
dkEGyyCLxqP6h/DU/tzNBdqFzetGtYvU4ftt3RO0a506cr2CHcdm8Q+/vPRJAAUR
tCFKb2huIEQuIFBvbHN0cmEgPGpkcEBwb2xzdHJhLmNvbT6JAJUDBRAzBNBE9RVb
+45ULV0BAWgiA/0WWO3+c3qlptPCHJ3DFm6gG/qNKsY94agL/mHOr0fxMP5l2qKX
O6a1bWkvGoYq0EwoKGFfn0QeHiCl6jVi3CdBX+W7bObMcoi+foqZ6zluOWBC1Jdk
WQ5/DeqQGYXqbYjqO8voCScTAPge3XlMwVpMZTv24u+nYxtLkE0ZcwtY9IkAlQMF
EDMEt/DHZvEPv7z0SQEBXh8D/2egM5ckIRpGz9kcFTDClgdWWtlgwC1iI2p9gEhq
aufy+FUJlZS4GSQLWB0BlrTmDC9HuyQ+KZqKFRbVZLyzkH7WFs4zDmwQryLV5wkN
C4BRRBXZfWy8s4+zT2WQD1aPO+ZsgRauYLkJgTvXTPU2JCN62Nsd8R7bJS5tuHEm
7HGmiQCVAwUQMwSvHB9/qQgDWPy9AQFAhAQAgJ1AlbKITrEoJ0+pLIsov3eQ348m
SVHEBGIkU3Xznjr8NzT9aYtq4TIzt8jplqP3QoV1ka1yYpZf0NjvfZ+ffYp/sIaU
wPbEpgtmHnVWJAebMbNs/Ad1w8GDvxEt9IaCbMJGZnHmfnEqOBIxF7VBDPHHoJxM
V31K/PIoYsHAy5w=
=cHFa
---END PGP PUBLIC KEY BLOCK---
```

29.2.8. Guido van Rooij <guido@FreeBSD.org>

```
Guido van Rooij <guido@gvr.win.tue.nl>
Fingerprint = 16 79 09 F3 C0 E4 28 A7  32 62 FA F6 60 31 C0 ED

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.2

mQCNAzGeO84AAAEEAKKAY91Na//DXwlUusr9GVESSlVwVP6DyH1wcZXhfN1fyZHq
SwhMCEdHYoojQds+VqD1iiZQvv1RLByBgj622PDAPN4+Z49HjGs7YbZsUNuQqPPU
wRPpP6ty69x1hPKq1sQIB5MS4radpCM+4wbZbhxv7l4rP3RWUbNaYutZnzI9AAUR
tCZHdWlkbyB2YW4gUm9vaWogPGd1aWRvQGd2ci53aW4udHVlLm5sPokAlQMFEDMG
Hcgff6kIA1j8vQEBbYgD/jm9xHuUuY+iXDkOzpCXBYACYEZDV913MjtyBAmaVqYo
Rh5HFimkGXe+rCo78Aau0hc57fFMTsJqnuWEqVt3GRq28hSK1FOZ7ni9/XibHcmN
rt2yugl3hYpClijo4nrDL1NxibbamkGW/vFGcljS0jqXz6NDVbGx5Oo7HBByxByz
iQCVAwUQMhmtVjt/x7zOdmsfAQFuVQQApsVUTigT5YWjQA9Nd5Z0+a/oVtZpyw5Z
OljLJP3vqJdMa6TidhfcatjHbFTve5x1dmjFgMX/MQTd8zf/+Xccy/PX4+lnKNpP
eSf1Y4aK+E8KHmBGd6GzX6CIboyGYLS9e3kGnN06F2AQtaLyJFgQ71wRaGuyKmQG
FwTn7jiKb1aJAJUDBRAyEOLXPt3iN6QQUSEBATwQA/9jqu0Nbk154+Pn+9mJX/YT
fYR2UqK/5FKCqgL5Nt/Deg2re0zMD1f8F9Dj6vuAAxq8hnOkIHKlWolMjkRKkzJi
mSPEWl3AuHJ31k948J8it4f8kq/o44usIA2KKVMlI63Q/rmNdfWCyiYQEVGcRbTm
GTdZIHYCOgV5dOo4ebFqgYkAlQMFEDIE1nMEJn15jgpJ0QEBW6kEAKqN8XSgzTqf
CrxFXT07MlHhfdbKUTNUoboxCGCLNW05vf1A8F5fdE5i14LiwkldWIzPxWD+Sa3L
fNPCfCZTaCiyGcLyTzVfBHA18MBAOOX6JiTpdcm22jLGUWBf/aJK3yz/nfbWntd/
LRHysIdVp29lP5BF+J9/Lzbb/9LxP1taiQCVAwUQMgRXZ44CzbsJWQz9AQFf7gP/
Qa2FS5S6RYKG3rYanWADVe/ikFV2lxuM1azlWbsmljXvKVWGe6cV693nS5lGGAjx
lbd2ADwXjlkNhv45HLWFm9PEveO9Jjr6tMuXVt8N2pxiX+1PLUN9CtphTIU7Yfjn
```

```
s6ryZZfwGHSfIxNGi5ua2SoXhg0svaYnxHxXmOtH24iJAJUDBRAyAkpV8qaAEa3W
TBkBARfQBAC+S3kbulEAN3SI7/A+A/dtl9DfZezT9C4SRBGsl2clQFMGIXmMQ/7v
7lLXrKQ7U2zVbgNfU8smw5h2vBIL6f1PyexSmc3mz9JY4er8KeZpcf6H0rSkHl+i
d7TF0GvuTdNPFO8hc9En+GG6QHOqbkB4NRZ6cwtfwUMhk2FHXBnjF4kAlQMFEDH5
FFukUJAsCdPmTQEBe74EAMBsxDnbD9cuI5MfF/QeTNEG4BIVUZtAkDme4Eg7zvsP
d3DeJKCGeNjiCWYrRTCGwaCWzMQk+/+MOmdkI6Oml+AIurJLoHceHS9jP1izdP7f
N2jkdeJSBsixunbQWtUElSgOQQ4iF5kqwBhxtOfEP/L9QsoydRMR1yB6WPD75H7V
iQCVAwUQMZ9YNGtaZ42Bsqd5AQH0PAQAhpVlAc3ZM/KOTywBSh8zWKVlSk3q/zGn
k7hJmFThnlhH1723+WmXE8aAPJi+VXOWJUFQgwELJ6R8jSU2qvk2m1VWyYSqRKvc
VRQMqT2wjss0GE1Ngg7tMrkRHT0il7E2xxIb8vMrIwmdkbTfYqBUhhGnsWPHZHq7
MoA1/b+rK7CJAJUDBRAxnvXh3IDyptUyfLkBAYTDA/4mEKlIP/EUX2Zmxgrd/JQB
hqcQlkTrBAaDOnOqe/4oewMKR7yaMpztYhJs97i03Vu3fgoLhDspE55ooEeHj0r4
cOdiWfYDsjSFUYSPNVhW4OSruMA3c29ynMqNHD7hpr3rcCPUi7J2RncocOcCjjK2
BQb/9IAUNeK4C9gPxMEZLokAlQMFEDGeO86zWmLrWZ8yPQEBEEID/2fPEUrSX3Yk
j5TJPFZ9MNX0lEo7AHYjnJgEbNI4pYm6C3PnMlsYfCSQDHuXmRQHAOWSdwOLvCkN
F8eDaF3M6u0urgeVJ+KVUnTz2+LZoZs12XSZKCte0HxjbvPpWMTTrYyimGezH79C
mgDVjsHaYOx3EXF0nnDmtXurGioEmW1J
=mSvM
---END PGP PUBLIC KEY BLOCK---
```

29.2.9. Peter Wemm <peter@FreeBSD.org>

```
Peter Wemm <peter@FreeBSD.org>
       aka <peter@spinner.dialix.com>
       aka <peter@haywire.dialix.com>
       aka <peter@perth.dialix.oz.au>
Key fingerprint = 47 05 04 CA 4C EE F8 93  F6 DB 02 92 6D F5 58 8A

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.3ia

mQCNAy9/FJwAAAEEALxs9dE9tFd0Ru1TXdq301KfEoe5uYKKuldHRBOacG2Wny6/
W3Ill57hOi2+xmq5X/mHkapywxvy4cyLdt31i4GEKDvxpDvEzAYcy2n9dIup/eg2
kEhRBX9G5k/LKM4NQsRIieaIEGGgCZRm0lINqw495aZYrPpO4EqGN2HYnOMZAAUT
tCVQZXRlciBXZW1tIDxwZXRlckBoYXl3aXJlLmRpYWxpeC5jb20+iQCVAwUQMwWT
cXW7bjh2o/exAQEFkQP+LIx5zKlYp1uR24xGApMFNrNtjh+iDIWnxxb2M2Kb6x4G
9z6OmbUCoDTGrX9SSL2Usm2RD0BZfyv9D9QRWC2TSOPkPRqQgIycc11vgbLolJJN
eixqsxlFeKLGEx9eRQCCbo3dQIUjc2yaOe484QamhsK1nL5xpoNWI1P9zIOpDiGJ
AJUDBRAxsRPqSoY3Ydic4xkBAbWLA/9q1Fdnnk4unpGQsG31Qbtr4AzaQD5m/JHI
4gRmSmbj6luJMgNG3fpO06Gd/Z7uxyCJB8pTst2a8C/ljOYZxWT+5uSzkQXeMi5c
YcI1sZbUpkHtmqPW623hr1PB3ZLA1TIcTbQW+NzJsxQ1Pc6XG9fGkT9WXQW3Xhet
AP+juVTAhLQlUGV0ZXIgV2VtbSA8cGV0ZXJAcGVydGguZGlhbGl4Lm96LmF1PokA
lQMFEDGxFCFKhjdh2JzjGQEB6XkD/2HOwfuFrnQUtdwFPUkgtEqNeSr64jQ3Maz8
```

```
xgEtbaw/ym1PbhbCk311UWQq4+izZE2xktHTFClJfaMnxVIfboPyuiSF99KHiWnf
/Gspet0S7m/+RXIwZi1qSqvAanxMiA7kKgFSCmchzas8TQcyyXHtn/gl9v0khJkb
/fv3R20btB5QZXRlciBXZW1tIDxwZXRlckBGcmVlQlNELm9yZz6JAJUDBRAxsRJd
SoY3Ydic4xkBAZJUA/4i/NWHz5LIH/R4IF/3V3LleFyMFr5EPFY0/4mcv2v+ju9g
brOEM/xd4LlPrx1XqPeZ74JQ6K9mHR64RhKR7ZJJ9A+12yr5dVqihe911KyLKab9
4qZUHYi36WQu2VtLGnw/t8Jg44fQSzbBF5q9iTzcfNOYhRkSD3BdDrC3llywO7Ql
UGV0ZXIgV2VtbSA8cGV0ZXJAc3Bpbm5lci5kaWFsaXguY29tPokAlQMFEDGxEi1K
hjdh2JzjGQEBdA4EAKmNFlj8RF9HQsoI3UabnvYqAWN5wCwEB4u+Zf8zq6OHic23
TzoK1SPlmSdBE1dXXQGS6aiDkLT+xOdeewNs7nfUIcH/DBjSuklAOJzKliXPQW7E
kuKNwy4eq5bl+j3HB27i+WBXhn6OaNNQY674LGaR41EGq44Wo5ATcIicig/z
=gv+h
---END PGP PUBLIC KEY BLOCK---
```

29.2.10. Jörg Wunsch <joerg@FreeBSD.org>

```
Type Bits/KeyID    Date       User ID
pub  1024/76A3F7B1 1996/04/27 Joerg Wunsch <joerg_wunsch@uriah.heep.sax.de>
            Key fingerprint = DC 47 E6 E4 FF A6 E9 8F  93 21 E0 7D F9 12 D6 4E
                              Joerg Wunsch <joerg_wunsch@interface-business.de>
                              Joerg Wunsch <j@uriah.heep.sax.de>
                              Joerg Wunsch <j@interface-business.de>

---BEGIN PGP PUBLIC KEY BLOCK---
Version: 2.6.3ia

mQCNAzGCFeAAAAEEAKmRBU2Nvc7nZy1Ouid61HunA/5hF4O91cXm71/KPaT7dskz
q5sFXvPJPpawwvqHPHfEbAK42ZaywyFp59L1GaYj87Pda+PlAYRJyY2DJl5/7JPe
ziq+7B8MdvbX6D526sdmcR+jPXPbHznASjkx9DPmK+7TgFujyXW7bjh2o/exAAUR
tC1Kb2VyZyBXdW5zY2ggPGpvZXJnX3d1bnNjaEB1cmlhaC5oZWVwLnNheC5kZT6J
AJUDBRA0FFkBs1pi61mfMj0BAfDCA/oCfkjrhvRwRCpSL8klJ1YDoUJdmw+v4nJc
pw3OpYXbwKOPLClsE7K3KCQscHel7auf91nrekAwbrXv9Clp0TegYeAQNjw5vZ9f
L6UZ5l3fH8E2GGA7+kqgNWs1KxAnG5GdUvJ9viyrWm8dqWRGo+loDWlZ12L2OgAD
fp7jVZTI1okAlQMFEDQPrLoff6kIA1j8vQEB2XQEAK/+SsQPCT/X4RB/PBbxUr28
GpGJMn3AafAaA3plYw3nb4ONbqEw9tJtofAn4UeGraiWw8nHYR2DAzoAjR6OzuX3
TtUV+57BIzrTPHcNkb6h8fPuHU+dFzR+LNoPaGJsFeov6w+Ug6qS9wa5FGDAgaRo
LHSyBxcRVoCbOEaS5S5EiQCVAwUQM5BktWVgqaw0+fnVAQGKPwP+OiWho3Zm2GKp
lEjiZ5zx3y8upzb+r1Qutb08jr2Ewja04hLg0fCrt6Ad3DoVqxe4POghIpmHM4O4
tcW92THQil70CLzfCxtfUc6eDzoP3krD1/Gwpm2hGrmYA9b/ez9+r2vKBbnUhPmC
glx5pf1IzHU9R2XyQz9Xu7FI2baOSZqJAJUDBRAyCIWZdbtuOHaj97EBAVMzA/41
VIph36l+yO9WGKkEB+NYbYOz2W/kyi74kXLvLdTXcRYFaCSZORSsQKPGNMrPZUoL
oAKxE25AoCgl5towqr/sCcu0A0MMvJddUvlQ2T+ylSpGmWchqoXCN7FdGyxrZ5zz
xzLIvtcio6kaHd76XxyJpltCASupdD53nEtxnu8sRrQxSm9lcmcgV3Vuc2NoIDxq
b2VyZ193dW5zY2hAaW50ZXJmYWNlLWJ1c2luZXNzLmRlPokAlQMFEDIIhfR1u244
```

```
dqP3sQEBWoID/RhBm+qtW+hu2fqAj9d8CVgEKJugrxZIpXuCKFvO+bCgQtogt9EX
+TJh4s8UUdcFkyEIu8CT2C3Rrr1grvckfxvrTgzSzvtYyv1072X3GkVY+SlUMBMA
rdl1qNW23oT7Q558ajnsaL065XJ5m7HacgTTikiofYG8i1s7TrsEeq6PtCJKb2Vy
ZyBXdW5zY2ggPGpAdXJpYWguaGVlcC5zYXguZGU+iQCVAwUQMaS91D4gHQUlG9CZ
AQGYOwQAhPpiobK3d/fz+jWrbQgjkoO+j39glYGXb22+6iuEprFRs/ufKYtjljNT
NK3B4DWSkyIPawcuO4Lotijp6jke2bsjFSSashGWcsJlpnwsv7EeFItT3oWTTTQQ
ItPbtNyLW6M6xB+jLGtaAvJqfOlzgO9BLfHuA2LY+WvbVW447SWJAJUDBRAxqWRs
dbtuOHaj97EBAXDBA/49rzZB5akkTSbt/gNd38OJgC+H8N5da25vV9dD3KoAvXfW
fw7OxIsxvQ/Ab+rJmukrrWxPdsC+1WU1+1rGa4PvJp/VJRDes2awGrn+iO7/cQoS
IVziC27JpcbvjLvLVcBIiy1yT/RvJ+87a3jPRHt3VFGcpFh4KykxxSNiyGygl4kA
lQMFEDGCUB31FVv7jlQtXQEB5KgD/iIJZe5lFkPr2B/Cr7BKMVBot1/JSu05NsHg
JZ3uK15w4mVtNPZcFi/dKbn+qRM6LKDFe/GF0HZD/ZD1FJt8yQjzF2w340B+F2GG
EOwnClqZDtEAqnIBzM/ECQQqH+6Bi8gpkFZrFgg5eON7ikqmusDnOlYStM/CBfgp
SbR8kDmFtCZKb2VyZyBXdW5zY2ggPGpAaW50ZXJmYWNlLWJ1c2luZXNzLmRlPokA
lQMFEDHioSdlYKmsNPn51QEByz8D/10uMrwP7MdaXnptd1XNFhpaAPYTVAOcaKlY
OGI/LLR9PiU3FbqXO+7INhaxFjBxa0Tw/p4au5Lq1+Mx81edHniJZNS8tz3I3goi
jIC3+jn2gnVAWnK5UZUTUVUn/JLVk/oSaIJNIMMDaw4J9xPVVkb+Fh1A+XqtPsVa
YESrNp0+iQCVAwUQMwXkzcdm8Q+/vPRJAQEA4QQAgNNX1HFgXrMetDb+w6yEGQDk
JCDAY9b6mA2HNeKLQAhsoZl4HwA1+iuQaCgo3lyFC+1Sf097OUTs74z5X1vCedqV
oFw9CxI3xuctt3pJCbbN68flOlnq0WdYouWWGlFwLlh5PEy//VtwX9lqgsizlhzi
t+fX6BT4BgKi5baDhrWJAJUDBRAyCKveD9eCJxX4hUkBAebMA/9mRPy6K6i7TX2R
jUKSl2p5oYrXPk12Zsw4ijuktslxzQhOCyMSCGK2UEC4UM9MXp1H1JZQxN/DcfnM
7VaUt+Ve0wZ6DC9gBSHJ1hKVxHe5XTj26mIr4rcXNy2XEDMK9QsnBxIAZnBVTjSO
LdhqqSMp3ULLOpBlRL2RYrqi27IXr4kAlQMFEDGpbnd1u244dqP3sQEBJnQD/RVS
Azgf4uorv3fpbosI0LE3LUufAYGBSJNJnskeKyudZkNkI5zGGDwVneH/cSkKT4OR
ooeqcTBxKeMaMuXPVl30QahgNwWjfuTvl5OZ8orsQGGWIn5FhqYXsKkjEGxIOBOf
vvlVQ0UbcR0N2+5F6Mb5GqrXZpIesn7jFJpkQKPU
=97h7
---END PGP PUBLIC KEY BLOCK---
```