How to create a port for OpenBSD- Basic2

Cómo crear un port para OpenBSD - Foremost

Autor: Fernando Quintero
Correo electrónico: fernando.a.quintero@gmail.com

Fecha de creación : 28/12/10 **Ultima modificación:** 20/01/11

Índice de contenido

1. Licencia (BSD)

2. Introducción

Al igual que en el documento Basic1[1], la motivación para crear este documento es la poca información existente sobre el proceso de creación de ports para OpenBSD. Estoy suponiendo un escenario donde el lector se quiere involucrar un poco mas con la comunidad y quiere hacerlo portando alguna aplicación para que funcione correctamente en OpenBSD. O quizás el lector desarrolló una aplicación y quiere que esta pueda funcionar en OpenBSD sin problemas. Portar una aplicación puede ser un proceso muy complejo, dependiendo de la aplicación por supuesto y se requieren unos conocimiento básicos de programación en diferentes lenguajes, por dar un ejemplo, lenguaje C, C++, perl y python.

Por otro lado, la persona que quiera aportar en este campo debe saber usar sistemas distribuidos para desarrollo de software, por ejemplo CVS, SVN, GIT, etc. No es necesario tener un conocimiento amplio, pero si por lo menos entender que es un commit, que es un patch o como publicar un “issue” en un sistema de tracking de bugs. Conocimientos avanzados en el uso del sistema operativo OBSD no sobrarán en este proceso.

Si el portador del software tiene experiencia en otros sistemas BSD o tipo Unix/Linux permitirá que se avance más rápido en esta entretenida tarea.

En este tutorial vamos a tomar como ejemplo el software **foremost,** un software usado para procesos de *datacarving*, procesos mediante el cual se logra extraer archivos de particiones o discos duros que pueden o no tener un formato, es una herramienta usada en procesos de análisis de evidencia en informática forense. Este software a diferencia del primer ejemplo, requiere que se creen parches para que funcione adecuadamente en OpenBSD, por eso este documento puede ser un poco (solo un poco) mas avanzado.

Toda la información relacionada con la creación de ports se puede encontrar en la lista de chequeo oficial: http://www.openbsd.org/checklist.html

1https://groups.google.com/group/openbsd-colombia/files

3. Preparando todo

Antes de empezar a trabajar es necesario tener un sistema OpenBSD instalado en versión current, esto significa que debemos usar el sistema CVS para actualizar el sistema base y el árbol de ports. En las guías oficiales donde se explica el proceso de portar software explican que es muy recomendable estar sincronizados en la versión del sistema operativo, esto se entiende mejor si pensamos en que el sistema operativo OpenBSD se compone de 4 grandes componentes:

- El kernel y todo lo referente al mismo
- Userland o todo el software que ejecuta el usuario
- El sistema gráfico
- y el sistema de ports

Estos 4 componentes deben estar sincronizados en sus versiones si queremos que no hayan problemas al momento de compilar nuestros ports, esto en resumen significa que si la versión del kernel y sus componentes se encuentran en *current,* entonces será necesario compilar las herramientas de userland para que estén en current también, lo mismo para el sistema gráfico y los ports. Volviendo al tema de la recomendación oficial es que cada uno de estos componentes se encuentren en current, pues es el escenario mas real donde se encuentra actualmente el desarrollo del sistema operativo y será el lugar donde podremos identificar la compatibilidad o no de nuestros ports respecto a todo el sistema como conjunto. Si quiere leer un poco mas sobre las diferentes versiones del sistema operativo y lo que esto implica para el usuario promedio, por favor busque el documento llamado "¿Qué es OpenBSD?" en el repositorio de documentos de la comunidad.

Una manera sencilla de tener nuestro sistema operativo en la versión current es descargando las versiones snapshots del sistema base (ISO instalable) y la versión snapshot del árbol de ports.

Una vez el sistema este instalado en versión current, podremos empezar con el proceso de creación de nuestro port. Si aún no ha leído los documentos básicos sobre el sistema de ports en OpenBSD, le aconsejamos hacerlo antes de continuar, pues es necesario estar familiarizado con la terminología para seguir este documento.

La razón del porque nos tomamos el tiempo para portar una aplicación puede variar de persona a persona, pero en general creo que es el deseo de colaborar y lograr que una aplicación que antes no estaba disponible ahora lo esté. Además, no es solamente que nosotros mismos podamos usar una aplicación especifica, para muchas personas tener un paquete disponible será una manera mas simple de usar buen software en OpenBSD, por lo tanto la creación de ports se entiende mas como un trabajo colaborativo donde varias personas se unen para hacer disponible cada vez mas software en nuestro sistema operativo favorito. Bienvenido!

4. Creando el port

La creación de un port es un proceso sistemático donde se debe intentar una y otra vez (compilar, empaquetar e instalar) hasta que todo funcione como debe funcionar, recuerde desde el principio que no solo es lograr instalar el software, se trata de hacerlo al estilo OpenBSD. Muchas veces los objetivos del software a instalar no están alineados con los objetivos del proyecto OpenBSD, me refiero a estructura, diseño, seguridad, etc. En estos casos lo mejor será asesorarnos con los desarrolladores en las listas oficiales del proyecto OpenBSD.

4.1. Paso 1: Preparandose para conseguir ayuda

El proceso de creación de ports no será un proceso solitario, lo primero que debe hacer es

inscribirse en la lista de correo ports@openbsd.org , esta lista esta diseñada para todas las personas que les gusta aportar en el proceso de creación y actualización de los ports.

Las listas disponibles[2] del proyecto OpenBSD se pueden encontrar en:

http://lists.openbsd.org/cgi-bin/mj_wwwusr?user=&passw=&func=lists-long

Seleccione la lista de ports y suscríbase a ella, haga los filtros adecuados para que separe los mensajes, pueden ser decenas durante el día.

Otra forma de conseguir ayuda es contactando directamente a los desarrolladores de OpenBSD (todos los que tengan @openbsd.org), en mi caso conozco un par de desarrolladores a los que les pido consejos directamente, son bastante amables siempre y cuando les hagas las preguntas correctas, en caso de no hacerlas recibirás un RTFM en distintos idiomas ;)

Uno de los que mas me ha apoyado en este aprendizaje y que actualmente mantiene mas de 200 ports es **Antoine Jacoutot** (ajacoutot@), puedes encontrar una entrevista a este gran profesional en:

http://www.undeadly.org/cgi?action=article&sid=20101018230848

Y bueno por último otra alternativa es apoyarse en las comunidades locales como OpenBSD Colombia, que aunque no somos expertos desarrolladores tenemos toda la intención de ayudar ;)

4.2. Paso 2: Seleccionar el paquete a portar

Si queremos realmente hacerle un seguimiento al port una vez creado, debemos seleccionar un software que usemos a diario y el cual nos interese actualizar constantemente. La tarea de portar un software no termina en el momento en que lo empaquetamos y esta listo para distribuir, podemos decir que esto solo es el comienzo, pues el verdadero trabajo esta en sus actualizaciones, parches y mejoras que podamos hacerle al software en si mismo. Muchas veces nos encontraremos con que las mejoras realizadas al paquete de OpenBSD son incorporadas al software original, obviamente se trata también de apoyar el desarrollo y no solo limitarnos a copiar y empaquetar.

En este caso y a modo de ejemplo vamos a portar el software *foremost* que se encuentra en:

http://foremost.sourceforge.net/

4.3. Paso 3

Creamos la estructura de directorios básica:

```
bash-4.1# mkdir foremost
bash-4.1# cd foremost/
bash-4.1# cp /usr/ports/infrastructure/templates/Makefile.template Makefile
bash-4.1# ls
bash-4.1# mkdir pkg
bash-4.1# touch pkg/DESCR
bash-4.1# touch pkg/PLIST
bash-4.1# pwd
/usr/ports/security/foremost
```

Se puede ver que el archivo Makefile aparece de una plantilla existente en el árbol de ports. Por la clase del software pensamos que debería estar en la categoría “net” o “security”.

2http://lists.openbsd.org/cgi-bin/mj_wwwusr?user=&passw=&func=lists-long

4.4. Paso 4

Lo primero que se debe hacer es crear una estructura básica dentro del archivo Makefile.

```
DIR     =     1.5
VERSION  =     ${DIR}.7
DISTNAME =     foremost-${VERSION}
PKGNAME  =     ${DISTNAME}
CATEGORIES =          security
COMMENT =             Foremost is a program to recover files based on their headers,
footers, and internal data structures

HOMEPAGE       =      http://foremost.sourceforge.net/pkg/

MAINTAINER =          fernando.a.quintero@gmail.com

PERMIT_PACKAGE_CDROM =  Yes
PERMIT_PACKAGE_FTP =    Yes
PERMIT_DISTFILES_CDROM =Yes
PERMIT_DISTFILES_FTP =  Yes

# where the source files and patches can be fetched
#

MASTER_SITES =        ${MASTER_SITE_SOURCEFORGE:=foremost/} \
                ${HOMEPAGE}

EXTRACT_SUFX =        .tar.gz
```

Las variables se auto explican, primero se pone la versión del software, luego el nombre que llevará el paquete en OpenBSD. Para la variable PKGNAME se debe mantener el formato *software-1.2.3* donde 1.2.3 es la versión del programa.

COMMENT es un comentario rápido del programa (una descripción rápida)

HOMEPAGE es el sitio web del proyecto, no necesariamente el lugar desde donde se descarga

MAINTAINER es la persona que se va a comprometer a tener el software portado a OpenBSD.

PERMIT_* las variables que definen si el software se puede redistribuir.

MASTER_SITES una variable que indica de donde se puede descargar el software, será usada cuando hagamos el *make fetch* como primera parte del proceso.

EXTRACT_SUFX variable que indica la extensión del archivo, necesaria para identificar el empaquetamiento original, también es usada en la ejecución del comando *make fetch.*

4.5. Paso 5

Una vez creado el archivo Makefile con estos datos básicos, empezamos con el proceso de ensayo y error hasta que el port se construya y se instale de forma correcta. Un port nunca se construirá en el primer intento o por lo menos no lo hará correctamente.

Empezamos descargando el archivo desde donde le configuramos usando el comando *make fetch*.

```
bash-4.1# make fetch
===>  Checking files for foremost-1.5.7
>> Fetch http://foremost.sourceforge.net/pkg/foremost-1.5.7.tar.gz
foremost-1.5.7.tar.gz 100% |
**********************************************************************************************************
**| 52352       00:00
>> Checksum file does not exist
```

Como se puede observar el archivo se descargó correctamente, pero nos informa que no hay un checksum para comprobar su integridad. En este punto es algo normal.

4.6. Paso 6: checksums

Creamos las sumas de comprobación de integridad (cheksums) para que cada vez que se descargue el port se valide su integridad en el sistema donde se piense instalar. Esto se hace con el comando *make checksum.*

```
bash-4.1# make makesum
===>  Checking files for foremost-1.5.7
`/usr/ports/distfiles/foremost-1.5.7.tar.gz' is up to date.

bash-4.1# cat distinfo
MD5 (foremost-1.5.7.tar.gz) = hgEZxJZlwqP7KwsdPbrQKg==
RMD160 (foremost-1.5.7.tar.gz) = Q34XTMzCXErmcoJaQTyGVVW7SkY=
SHA1 (foremost-1.5.7.tar.gz) = wm1omQ171SRdX33IPJIXZCp6IFY=
SHA256 (foremost-1.5.7.tar.gz) =
UCBU7yEuPZCykumcf3rJH4nwJHIM1afnaAw9GQHvXzQ=
SIZE (foremost-1.5.7.tar.gz) = 52352
bash-4.1#
```

El archivo *distinfo* es creado automáticamente con la información de diferentes hashes.

4.7. Paso 7: extract

Hacemos una extracción del software, esto significa que se desempaqueta y se descomprime en un directorio especial determinado por el sistema de ports. El comando a usar es *make extract.*

```
bash-4.1# make extract
===>  Checking files for foremost-1.5.7
`/usr/ports/distfiles/foremost-1.5.7.tar.gz' is up to date.
>> (SHA256) foremost-1.5.7.tar.gz: OK
===>  Extracting for foremost-1.5.7
```

Si queremos verificar que el programa se descomprime correctamente, debemos encontrar la ruta donde quedó almacenado por el sistema de ports, esto se hace con el siguiente comando:

```
bash-4.1# make show=WRKDIST
/usr/ports/pobj/foremost-1.5.7/foremost-1.5.7
```

4.8. Paso 8

Parchamos el software, este paso se realiza para aplicarle los parches necesarios al software para que pueda compilar, construir (build) e instalar correctamente en el sistema. Por defecto no tendremos parches, mas adelante explicaremos como y porque crearlos.

Para aplicar los parches, usamos el comando *make patch*.

```
bash-4.1# make patch
===>  Patching for foremost-1.5.7
bash-4.1#
```

4.9. Paso 9

Llegamos al proceso de la compilación, en este paso ejecutaremos el comando make build el cual se encargará de compilar el software y construir sus dependencias, sin embargo no podemos esperar a que sea un proceso mágico, así que debemos tener en cuenta la forma como esta construido el Makefile del código fuente del programa antes de intentar compilarlo con el sistema de ports.,

Antes de este paso es necesario entender que el proceso general de la creación de un port es el siguiente: *fetch → extract → build → fake → install*

Explico los pasos a continuación:

fetch → se encarga de descargar el software

extract → se encarga de descomprimirlo en un directorio especifico

build → se encarga de compilar el software y de organizarlo para instalarlo

fake → esta es una opción especial que se usa para que el software se instale en un directorio temporal (fake) antes de ser instalado en el sistema real, de esta forma OBSD previene que corrompamos el sistema "jugando" con los ports.

Install → una vez comprobado el fake, podemos hacer la instalación real en el sistema y generalmente todos los archivos creados por el port deben quedar bajo el directorio */usr/local/*

Existen pasos intermedios que se pueden aprovechar para poner instrucciones que necesitemos, por ejemplo existe un *pre-install, do-install y pos-install*. En este caso vamos a ejecutar el comando make build y analizaremos lo que sucede.

```
bash-4.1# make build
===>  Patching for foremost-1.5.7
===>  Configuring for foremost-1.5.7
===>  Building for foremost-1.5.7
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c main.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c state.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c helpers.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c config.c
config.c: In function 'translate':
config.c:27: warning: value computed is not used
config.c:32: warning: value computed is not used
config.c:37: warning: value computed is not used
config.c:42: warning: value computed is not used
config.c:47: warning: value computed is not used
config.c:52: warning: value computed is not used
config.c:57: warning: value computed is not used
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c cli.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c engine.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c dir.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c extract.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX -c api.c
gcc -Wall -O2 -DVERSION=\"1.5.7\" -D__UNIX main.o state.o helpers.o config.o cli.o
engine.o dir.o extract.o api.o -o foremost
state.o(.text+0x278): In function `init_builtin':
: warning: strcpy() is almost always misused, please use strlcpy()
engine.o(.text+0xcb2): In function `search_chunk':
: warning: sprintf() is often misused, please use snprintf()
dir.o(.text+0x7e7): In function `create_sub_dirs':
: warning: strcat() is almost always misused, please use strlcat()
bash-4.1#
```

4.10. Paso 10

Al parecer el software se compiló adecuadamente. Sigamos entonces con el procedimiento, hagamos un *make fake*.

```
bash-4.1# make fake
===>  Faking installation for foremost-1.5.7
install -m 755 foremost /usr/local/bin
install -m 444 foremost.8.gz /usr/share/man/man8
install -m 444 foremost.conf /usr/local/etc
```

4.11. Paso 11

Como no hubo errores sigamos con el comando *make plist*, este se encargará de crear el listado de todos los archivos y las rutas que se crearan en el sistema cuando el paquete sea instalado.

```
bash-4.1# make plist
===>  Updating plist for foremost-1.5.7
Scanning destdir
Getting old lists
1st pass identifying files
Attaching annotations
Sorting out destdir files
pkg/PLIST empty
bash-4.1# cat pkg/PLIST
@comment $OpenBSD$
bash-4.1#
```

En este punto podemos ver que hay un error, pues el archivo PLIST no contiene archivos a instalar, esto ocurrió porque en el paso 10 el sistema instaló directamente los binarios del paquete en las rutas del sistema en vez de hacerlo en el directorio fake, recordemos que el directorio fake se usa como un paso intermedio para verificar que el paquete se instala y desinstala adecuadamente, para garantizar que un port no presentará problemas es necesario verificar que la instalación con el make fake se haga correctamente. **¿Qué hacemos entonces?**

4.12. Paso 12: SYSTRACE

Vamos a agregar una variable al archivo Makefile que nos permitirá identificar posibles errores durante los pasos *make build* y *make fake*.

La variable funciona verificando que ningún archivo del software que se esta portando se instale directamente en una ruta del sistema. La variable se llama USE_SYSTRACE y debe estar definida con el valor Yes como se ve a continuación.

```
USE_SYSTRACE   = Yes
```

Para intentarlo de nuevo ejecutamos los siguientes comandos:

```
-bash-4.1#make clean=all
-bash-4.1#make uninstall
```

Estos comandos se encargan de limpiar el proceso que hemos realizado hasta el momento con el objetivo de que repitamos los pasos build y fake. No se preocupe, el archivo Makefile no se eliminará. Ejecutamos entonces el comando *make build* nuevamente y cuando intentamos correr el comando *make fake*, nos encontramos con lo siguiente:

```
===>  Faking installation for foremost-1.5.7
install -m 755 foremost /usr/local/bin
systrace: deny user: root, prog: /usr/bin/install, pid: 19036(0)[3368], policy:
/usr/bin/env, filters: 197, syscall: native-fswrite(10), filename:
/usr/local/bin/foremost
systrace: deny user: root, prog: /usr/bin/install, pid: 19036(0)[3368], policy:
/usr/bin/env, filters: 197, syscall: native-fswrite(5), filename:
/usr/local/bin/foremost
install: /usr/local/bin/foremost: Operation not permitted
*** Error code 71

Stop in /usr/ports/pobj/foremost-1.5.7/foremost-1.5.7 (line 121 of Makefile).
*** Error code 1

Stop in /usr/ports/security/foremost (line 2532 of
/usr/ports/infrastructure/mk/bsd.port.mk).
*** Error code 1

Stop in /usr/ports/security/foremost (line 2232 of
/usr/ports/infrastructure/mk/bsd.port.mk).
-bash-4.1#
```

Podemos ver algunos errores, esto se debe a que el systrace identifico que el port se quiere instalar directamente en el sistema por eso nos lanza el error y no nos dejará pasar hasta que no lo solucionemos. Pero como se soluciona?, aplicando parches a las rutas donde deben ser instalados los componentes del software que se esta portando.

4.13. Paso 13: patch

El procedimiento la mayoría de las veces consiste en ir al directorio donde se extrajo el software y verificar el archivo Makefile del código fuente para entender lo que trata de hacer el programa al momento de instalarse. Una vez identificadas estas rutas podremos generar un parche que arregle el problema en el momento de compilación. Veamos.

Ingresamos al directorio /usr/ports/pobj/foremost-1.5.7/foremost-1.5.7 y sacamos una copia del archivo Makefile (el original del software, no el del port), la copia debe llamarse con el mismo nombre y terminada en la extensión .orig. Este procedimiento se repetirá para cada archivo del que queramos generar un parche.

-bash-4.1#cp Makefile Makefile.orig

Luego vamos a editar las siguientes líneas:

```
BIN = ${PREFIX}/usr/local/bin
MAN = ${PREFIX}/usr/share/man/man8
CONF= ${PREFIX}/usr/local/etc
```

Para convertirlas en:

```
BIN = ${PREFIX}/usr/local/bin
MAN = ${PREFIX}/usr/share/man/man8
CONF= ${PREFIX}/usr/local/etc
```

Esto se observa mejor sacando un archivo diff (un archivo parche) que nos muestra las diferencias entre lo que había y lo que quedó. Los símbolos + significan cosas que se agregaron y el símbolo

– significan las cosas que se eliminaron, para mas información busque ayuda en el manual del comando diff [3]

```
bash-4.1# diff -uNp Makefile.orig Makefile
--- Makefile.orig       Mon Dec 20 01:44:36 2010
+++ Makefile    Mon Dec 20 01:45:51 2010
@@ -24,9 +24,9 @@ MAN_PAGES = $(NAME).8.gz
RAW_FLAGS += -DVERSION=\"$(VERSION)\"

# Where we get installed
-BIN = /usr/local/bin
-MAN = /usr/share/man/man8
-CONF= /usr/local/etc
+BIN = ${PREFIX}/usr/local/bin
+MAN = ${PREFIX}/usr/share/man/man8
+CONF= ${PREFIX}/usr/local/etc
# Setup for compiling and cross-compiling for Windows
# The CR_ prefix refers to cross compiling from OSX to Windows
CR_CC = $(CR_BASE)/gcc
bash-4.1#
```

Como se puede ver, quitamos 3 líneas y agregamos 3, antes de la ruta se puso la variable ${PREFIX} que lo que hacen es mantenerlo dentro del directorio fake.

${PREFIX} es una variable de la estructura del árbol de ports que apunta al directorio fake, por eso es necesario indicarle al programa que su nueva ruta de instalación será dentro del fake. Algo bonito de esta estructura de los ports es que aunque el programa se instale en una ruta diferente (fake) al momento de instalar el paquete construido con este árbol de ports, el mismo se instalará en las rutas adecuadas dentro del sistema, esto es, se instalará en los directorios /bin, /usr/local/bin o donde corresponda, así que el asunto del fake solo aplica cuando estamos en el proceso de construcción del port.

4.14. Paso 14

Una vez creado el archivo Makefile.orig y después de haber hecho los cambios en el Makefile real, procedemos a construir el parche de forma automática evitando el procedimiento manual como lo vimos en el ejemplo con el comando diff. El comando que necesitamos es *make update-patches*.

```
bash-4.1# pwd
/usr/ports/security/foremost
bash-4.1# make update-patches
Processing Makefile
No patch-* found for Makefile, creating patch-Makefile
edit patches:
```

El parche se crea automáticamente en el directorio patches y usa el nombre patch seguido del nombre del archivo modificado. En caso de que el archivo tenga un nombre que contenga espacios o puntos, estos serán reemplazados por el caracter _.

3 http://www.openbsd.org/cgi-bin/man.cgi?query=diff&apropos=0&sektion=0&manpath=OpenBSD+Current&arch=i386&format=html

```
bash-4.1# cat patches/patch-Makefile
$OpenBSD$
--- Makefile.orig       Mon Dec 20 01:44:36 2010
+++ Makefile    Mon Dec 20 01:45:51 2010
@@ -24,9 +24,9 @@ MAN_PAGES = $(NAME).8.gz
RAW_FLAGS += -DVERSION=\"$(VERSION)\"

# Where we get installed
-BIN = /usr/local/bin
-MAN = /usr/share/man/man8
-CONF= /usr/local/etc
+BIN = ${PREFIX}/usr/local/bin
+MAN = ${PREFIX}/usr/share/man/man8
+CONF= ${PREFIX}/usr/local/etc
# Setup for compiling and cross-compiling for Windows
# The CR_ prefix refers to cross compiling from OSX to Windows
CR_CC = $(CR_BASE)/gcc
bash-4.1#
```

Cuando intentamos ejecutar el comando *make fake*, nuevamente obtenemos un error:

```
 warning: sprintf() is often misused, please use snprintf()
dir.o(.text+0x7e7): In function `create_sub_dirs':
: warning: strcat() is almost always misused, please use strlcat()
===>  Faking installation for foremost-1.5.7
install -m 755 foremost /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/usr/local/bin
install: /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/usr/local/bin: No such file or
directory
*** Error code 71

Stop in /usr/ports/pobj/foremost-1.5.7/foremost-1.5.7 (line 121 of Makefile).
*** Error code 1

Stop in /usr/ports/security/foremost (line 2532 of
/usr/ports/infrastructure/mk/bsd.port.mk).
*** Error code 1

Stop in /usr/ports/security/foremost (line 2232 of
/usr/ports/infrastructure/mk/bsd.port.mk).
bash-4.1#
```

De este error se puede evidenciar que hay un problema con la ruta al binario, esta queda duplicada de la forma **/usr/local/bin/usr/local/bin/** esto significa que el parche quedó mal aplicado, como lo mencionaba al principio del documento es algo normal, seguramente vamos a tener que aplicar una y otra vez diferentes parches hasta que todo concuerde como se espera.

4.15. Paso 15

Para generar nuevamente el parche repetimos el paso 14 y obtenemos el nuevo parche:

```
bash-4.1# cat patches/patch-Makefile
$OpenBSD$
--- Makefile.orig       Mon Dec 20 01:44:36 2010
+++ Makefile    Mon Dec 20 01:45:51 2010
@@ -24,9 +24,9 @@ MAN_PAGES = $(NAME).8.gz
RAW_FLAGS += -DVERSION=\"$(VERSION)\"

# Where we get installed
-BIN = /usr/local/bin
-MAN = /usr/share/man/man8
-CONF= /usr/local/etc
+BIN = ${PREFIX}/bin
+MAN = ${PREFIX}/share/man/man8
+CONF= ${PREFIX}/etc
# Setup for compiling and cross-compiling for Windows
# The CR_ prefix refers to cross compiling from OSX to Windows

CR_CC = $(CR_BASE)/gcc
bash-4.1#
```

Construido este parche seguimos con el fake, ejecutando el comando *make fake*, recuerde que este comando lo que hace es instalar en un directorio temporal todo el software, este paso es necesario para verificar que el software se instalará bien en el futuro y no habrá problemas de dependencias, rutas equivocadas o permisos incorrectos. Una vez ejecutado el comando obtenemos lo siguiente:

```
===>  Faking installation for foremost-1.5.7
install -m 755 foremost /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/bin
install -m 444 foremost.8.gz /usr/ports/pobj/foremost-1.5.7/fake-
i386/usr/local/share/man/man8
install: /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/share/man/man8: No such file or
directory
```

Al parecer la ruta del binario quedo bien instalada, sin embargo la página del manual no queda en el lugar correcto para OpenBSD. ¿Qué hacemos entonces?. Adivinaste!, mejoramos el parche repitiendo el procedimiento del paso 14 y volvemos a ejecutar el *make fake.*

Nuevo parche:

```
bash-4.1# cat patches/patch-Makefile
$OpenBSD$
--- Makefile.orig       Mon Dec 20 01:44:36 2010
+++ Makefile    Mon Dec 20 01:45:51 2010
@@ -24,9 +24,9 @@ MAN_PAGES = $(NAME).8.gz
RAW_FLAGS += -DVERSION=\"$(VERSION)\"

# Where we get installed
-BIN = /usr/local/bin
-MAN = /usr/share/man/man8
-CONF= /usr/local/etc
+BIN = ${PREFIX}/bin
+MAN = ${PREFIX}/man/man8
+CONF= ${PREFIX}/etc
# Setup for compiling and cross-compiling for Windows
# The CR_ prefix refers to cross compiling from OSX to Windows
CR_CC = $(CR_BASE)/gcc
bash-4.1#
```

Resultado de ejecutar *make fake*:

```
===>  Faking installation for foremost-1.5.7
install -m 755 foremost /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/bin
install -m 444 foremost.8.gz /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/man/man8
install -m 444 foremost.conf /usr/ports/pobj/foremost-1.5.7/fake-i386/usr/local/etc
bash-4.1#
```

Al parecer la instalación en el directorio fake es correcta, Qué hacemos entonces?

¿Intentamos construir el paquete?. NO!.

Sería posible construir el paquete de esta forma, sin embargo en OpenBSD hay unas normas claras sobre donde poner algunos archivos de configuración, en este caso el archivo de ejemplo del foremost debería estar en la ruta: **/usr/local/share/examples/foremost**

Así que agregamos al Makefile del port las siguientes líneas:

```
pre-install:
	${INSTALL_DATA_DIR} ${PREFIX}/share/examples/foremost
```

Estas líneas le dirán al port que antes de empezar a construirlo debe crear dicho directorio.

En este punto el archivo Makefile del port debe ser:

```
bash-4.1# cat Makefile
DIR	=	1.5
VERSION	=	${DIR}.7
DISTNAME =	foremost-${VERSION}
PKGNAME	=	${DISTNAME}
CATEGORIES =		security
COMMENT =		Foremost is a program to recover files based on their headers, footers, and
internal data structures

HOMEPAGE	=	http://foremost.sourceforge.net/pkg/

MAINTAINER =		fernando.a.quintero@gmail.com

PERMIT_PACKAGE_CDROM =	Yes
PERMIT_PACKAGE_FTP =	Yes
PERMIT_DISTFILES_CDROM =Yes
PERMIT_DISTFILES_FTP =	Yes

# where the source files and patches can be fetched
#

MASTER_SITES =		${HOMEPAGE}

EXTRACT_SUFX =		.tar.gz

# "make port-lib-depends-check" can help
#WANTLIB =		???

USE_SYSTRACE	=	Yes

pre-install:
	${INSTALL_DATA_DIR} ${PREFIX}/share/examples/foremost

.include <bsd.port.mk>
bash-4.1#
```

Este archivo funciona siempre y cuando cambiemos nuevamente el parche para que el archivo de configuración de ejemplo quede en la ruta recién creada.

El parche final es entonces:

```
bash-4.1# cat patches/patch-Makefile
$OpenBSD$
--- Makefile.orig       Mon Dec 20 05:27:44 2010
+++ Makefile    Mon Dec 20 05:27:58 2010
@@ -24,9 +24,9 @@ MAN_PAGES = $(NAME).8.gz
RAW_FLAGS += -DVERSION=\"$(VERSION)\"

# Where we get installed
-BIN = /usr/local/bin
-MAN = /usr/share/man/man8
-CONF= /usr/local/etc
+BIN = ${PREFIX}/bin
+MAN = ${PREFIX}/man/man8
+CONF= ${PREFIX}/share/examples/foremost
# Setup for compiling and cross-compiling for Windows
# The CR_ prefix refers to cross compiling from OSX to Windows
CR_CC = $(CR_BASE)/gcc
bash-4.1#
```

4.16. Paso 16

Después de haber hecho el *make fake,* podemos hacer un *make plist* para generar un archivo PLIST donde estarán las rutas y archivos que se crearán en el sistema cuando se instale el port, este archivo también servirá para eliminar directorios y archivos instalados por el port.

```
bash-4.1# make update-plist
===>  Updating plist for foremost-1.5.7
Scanning destdir
Getting old lists
1st pass identifying files
Attaching annotations
Sorting out destdir files
pkg/PLIST changed
pkg/PLIST.orig present
*** Error code 1

Stop in /usr/ports/security/foremost (line 2193 of
/usr/ports/infrastructure/mk/bsd.port.mk).
*** Error code 1

Stop in /usr/ports/security/foremost (line 2232 of
/usr/ports/infrastructure/mk/bsd.port.mk).
bash-4.1#
```

Como se puede ver el comando no se ejecuta con éxito porque ya existe un archivo pkg/PLIST.orig (esto sucede cuando ejecutamos varias veces make plist), la solución es borrar el archivo existente para que el comando lo pueda regenerar.

```
bash-4.1# rm pkg/PLIST*
bash-4.1# make update-plist
===>  Updating plist for foremost-1.5.7
Scanning destdir
Getting old lists
1st pass identifying files
Attaching annotations
Sorting out destdir files
pkg/PLIST is new
bash-4.1# cat pkg/PLIST
@comment $OpenBSD$
@bin bin/foremost
@man man/man8/foremost.8.gz
share/examples/foremost/
share/examples/foremost/foremost.conf
bash-4.1#
```

4.17. Paso 17: creando el paquete

Superada la ejecución del comando *make plist,* procedemos a crear el paquete con el comando *make package*:

```
bash-4.1# make package
`/usr/ports/pobj/foremost-1.5.7/fake-i386/.fake_done' is up to date.
===>  Building package for foremost-1.5.7
Create /usr/ports/packages/i386/all/foremost-1.5.7.tgz
Fatal error: can't parse Fatal error: comment is too long
Foremost is a program to recover files based on their headers, footers, and internal data
structures
                                        ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

at /usr/libdata/perl5/OpenBSD/PkgCreate.pm line 618
*** Error code 1
```

Se puede ver que se genera un error porque la descripción del port es demasiado larga, la solución es acortarla y volver a ejecutar el comando.

```
bash-4.1# make package
`/usr/ports/pobj/foremost-1.5.7/fake-i386/.fake_done' is up to date.
===>  Building package for foremost-1.5.7
Create /usr/ports/packages/i386/all/foremost-1.5.7.tgz
Link to /usr/ports/packages/i386/ftp/foremost-1.5.7.tgz
Link to /usr/ports/packages/i386/cdrom/foremost-1.5.7.tgz
bash-4.1#
```

Esta vez el comando se ejecuta sin problemas y nos crea los tan esperando .tgz, que corresponden a los paquetes instalábles de OpenBSD.

4.18. Paso 18: install

Instalamos el paquete recién creado:

```
bash-4.1# make install
===>  Installing foremost-1.5.7 from /usr/ports/packages/i386/all/
foremost-1.5.7: ok
bash-4.1#
```

Con esto el paquete queda correctamente instalado, pero al intentar ejecutarlo nos damos cuenta que busca un archivo de configuración en una ruta especifica: **/etc/foremost/foremost,conf**, ¿Cómo podemos entonces arreglar el paquete para que esta ruta este funcional una vez instalado el paquete?

La solución es editar directamente el archivo PLIST (algunas veces será necesario hacerlo) y agregarle las rutas donde queremos tener una copia del archivo de ejemplo que ya instala el paquete. Mas información sobre como trabajar con archivos de configuración y similares en http://www.openbsd.org/porting/config.html.

El archivo PLIST final queda entonces de la siguiente forma:

```
bash-4.1# pwd
/usr/ports/security/foremost
bash-4.1# cat pkg/PLIST
@comment $OpenBSD$
@bin bin/foremost
@man man/man8/foremost.8.gz
share/examples/foremost/
@sample ${SYSCONFDIR}/foremost/
share/examples/foremost/foremost.conf
@sample ${SYSCONFDIR}/foremost/foremost.conf
bash-4.1#
```

Como se puede ver el “comando” @ se usa debajo de la línea que queramos usar, por ejemplo antes de copiar el archivo necesitamos crear el directorio, por eso usamos una línea con arroba (@) inmediatamente debajo de la línea que crea el directorio.

share/examples/foremost/
@sample ${SYSCONFDIR}/foremost/

Luego el archivo de ejemplo lo copiaremos en la ruta que esperamos que sea /etc/foremost/ :

share/examples/foremost/foremost.conf
@sample ${SYSCONFDIR}/foremost/foremost.conf

4.19. Paso 19

Ya para finalizar debemos asegurarnos que el port instala y desinstala sin problemas, por eso hacemos un par de pruebas mas:

Primero lo desinstalamos y borramos todo lo que se genero en los temporales para volver a empezar:

```
bash-4.1# make uninstall
===> Deinstalling for foremost-1.5.7
foremost-1.5.7: ok
Read shared items: ok
```

```
bash-4.1# make clean=all
===>  Cleaning for foremost-1.5.7
rm -f /usr/ports/packages/i386/all/foremost-1.5.7.tgz
/usr/ports/packages/i386/ftp/foremost-1.5.7.tgz /usr/ports/packages/i386/cdrom/foremost-
1.5.7.tgz
cd /usr/ports/plist/i386 && rm -f foremost-1.5.7
bash-4.1#
```

Luego hacemos el procedimiento de instalación nuevamente, ejecutando *make install*.

Una vez instalado el programa cuando intentamos ejecutarlo observamos que el archivo de configuración aún no se encuentra, así que algo nos esta faltando. Analizando un poco los archivos fuentes, podemos encontrar que existen lugares donde de forma *harcoded* se indica en donde estará el archivo de configuración, como el cambio que queremos hacer requiere modificar los archivos fuentes, será necesario crear un nuevo parche.

El cambio lo hacemos en el archivo *config.c* y ejecutamos el comando *make update-patches* el cual creará el archivo resultante que debe quedar similar a este:

```
$OpenBSD$
--- config.c.orig       Mon Dec 20 06:06:33 2010
+++ config.c    Mon Dec 20 06:07:26 2010
@@ -288,7 +288,7 @@ int load_config_file(f_state *s)
#ifdef __WIN32
            set_config_file(s, "/Program Files/foremost/foremost.conf");
#else
-            set_config_file(s, "/usr/local/etc/foremost.conf");
+            set_config_file(s, "/etc/foremost/foremost.conf");
#endif
            if ((f = fopen(get_config_file(s), "r")) == NULL)
                  {
~
bash-4.1# ls patches/
patch-Makefile patch-config_c
bash-4.1#
```

El Makefile final del port queda de la siguiente forma:

```
-bash-4.1# cat Makefile
DIR      =      1.5
VERSION  =      ${DIR}.7
DISTNAME =      foremost-${VERSION}
PKGNAME  =      ${DISTNAME}
CATEGORIES =            security
COMMENT =               Foremost is a program to recover files doing datacarving

HOMEPAGE        =       http://foremost.sourceforge.net/pkg/

MAINTAINER =            fernando.a.quintero@gmail.com

PERMIT_PACKAGE_CDROM =  Yes
PERMIT_PACKAGE_FTP =    Yes
PERMIT_DISTFILES_CDROM =Yes
PERMIT_DISTFILES_FTP =  Yes

MASTER_SITES =          ${HOMEPAGE}

EXTRACT_SUFX =          .tar.gz

USE_SYSTRACE    =       Yes

pre-install:
      ${INSTALL_DATA_DIR} ${PREFIX}/share/examples/foremost

.include <bsd.port.mk>
-bash-4.1#
```

4.20. Paso 20: chequeo de dependencias

Ahora si para terminar debemos garantizar que el port no requiera otras librerías al momento de instalarse porque entonces quedará instalado como un paquete roto, para esto la infraestructura de ports nos brinda el comando *make port-lib-depends-check*.

```
foremost-1.5.7(security/foremost):
WANTLIB:       c.58 (/usr/local/bin/foremost) (system lib)
      WANTLIB += c
*** Error code 1 (ignored)
```

Después de ejecutar este comando veremos un error indicandonos que nuestro port requiere de la librería c, por lo tanto editamos por última vez el archivo Makefile del port para agregarle la dependencia. La línea se agrega como lo sugiere la salida del comando:

```
WANTLIB += c
```

Una vez agregada la dependencia volvemos a ejecutar el comando *make port-lib-depends-check* y verificamos que no se genere ningún error, es posible que tengamos que agregar varias líneas para suplir todas las dependencias.

4.21. Paso 21

Es necesario que antes de enviar el port a la lista @ports o compartirlo con alguien es necesario recortar a 72 columnas la longitud de las columnas del archivo /pkg/DESCR.

El comando para esto es:

```
-bash-4.1# fmt -w 72 DESCR
Foremost is a console program to recover files based on their headers,
footers, and internal data structures. This process is commonly referred
to as data carving.  Foremost can work on image files, such as those
generated by dd, Safeback, Encase, etc, or directly on a drive. The
headers and footers can be specified by a configuration file or you can
use command line switches to specify built-in file types. These built-in
types look at the data structures of a given file format allowing for a
more reliable and faster recovery.
-bash-4.1#
```

4.22. Final final

Por fin!, hemos logrado crear nuestro port, ahora podemos enviarlo a las listas de correo para que las personas que lo requieran lo puedan usar o podemos subirlo a algún sitio web, con toda seguridad alguien le agradecerá el esfuerzo de portar una aplicación.

El archivo Makefile final se muestra a continuación;

```
-bash-4.1# cat Makefile
DIR      =      1.5
VERSION  =      ${DIR}.7
DISTNAME =      foremost-${VERSION}
PKGNAME  =      ${DISTNAME}
CATEGORIES =            security
COMMENT =               Foremost is a program to recover files doing datacarving
HOMEPAGE        =       http://foremost.sourceforge.net/pkg/
MAINTAINER =            fernando.a.quintero@gmail.com

PERMIT_PACKAGE_CDROM =  Yes
PERMIT_PACKAGE_FTP =    Yes
PERMIT_DISTFILES_CDROM =Yes
PERMIT_DISTFILES_FTP =  Yes

MASTER_SITES =          ${HOMEPAGE}
EXTRACT_SUFX =          .tar.gz
WANTLIB =               c
USE_SYSTRACE    =       Yes

pre-install:
      ${INSTALL_DATA_DIR} ${PREFIX}/share/examples/foremost

.include <bsd.port.mk>
-bash-4.1#
```

5. Contacto

Recuerde que para las preguntas relacionadas con el tema del sistema de ports o cualquier otro tema relacionado con OpenBSD puede consultar las listas oficiales de correo en ingles o contactar a las personas de la comunidad OpenBSD Colombia que ofrecen documentación y soporte totalmente en español. Dudas especificas o aclaraciones directamente al correo del autor.

Espero que este tutorial haya sido de su agrado.

6. Tips durante la construcción de ports

- Buscar el port equivalente en cualquiera de los otros sistemas BSD existentes, con el fin de tener una idea de como debe quedar construido el archivo Makefile, por ejemplo en sistemas FreeBSD o NetBSD.
- Intentar compilar el software sin usar el sistema de ports, puede ser bajo el directorio /tmp o /root, el proceso debería seguir las indicaciones de los archivos README o INSTALL que acompañe el software original. Compilarlo de esta forma puede ser una tarea complicada pero le dará ideas claras sobre lo que se requiere en un archivo Makefile.
- Buscar el mismo port en la lista oficial de OpenBSD, muchas veces sucede que alguien esta trabajando en el port o ya lo han enviado a la lista, así que es cuestión de retomar el trabajo y avanzar. Como ejemplo puntual para el software foremost encontramos este email enviado a la lista hace algún tiempo: http://marc.info/?t=128056352800005&r=1&w=1
- Buscar el port en el sitio oficial del upstream (desarrollador oficial/original del software) y tratar de probarlo o estudiarlo antes de empezar a portarlo.
- Buscar en google errores de compilación que se generen durante el *make build* o *make fake* del port. Muchas veces nos dará pistas de como seguir adelante.
- Buscar ports similares en las rutas */usr/ports/*, de esta forma podemos hacernos la idea de como construirlo o aprendernos algún truco usado por otro portador de software.

7. Enlaces relacionados

- En los archivos oficiales de la comunidad OpenBSD Colombia, puedes encontrar documentos que pueden apoyarte en la creación de nuevos ports.

  https://groups.google.com/group/openbsd-colombia/files

  http://www.openbsdcolombia.org/documentos/

- Entrevista a Marc @espie desarrollador de OpenBSD:

  http://mongers.org/openbsd/interview-espie-ports

- Como crear un port:
  http://www.openbsd.org/papers/opencon07-portstutorial/

- Como crear un port avanzado:
  http://www.openbsd.org/papers/capbug200712/

- Como crear un port de forma rápida:
  http://www.undeadly.org/cgi?action=article&sid=20080318060000

! Larga vida a OpenBSD !