はじめに

Solaris で pkgsrc 使うときのめも。といっても、実際試しているのは、Solaris 8 だけだけどね。

- [2005/4/4] 追記：Solaris をさわる環境が一切なくなってしまったので、このページは（たぶん）更新されません。（また環境が手に入れば、更新を再開したいと思います。）
- [2007/2/20] 追記：最近の環境に合うように若干修正しました。

なぜ pkgsrc なの？

Solaris へのサードパーティ製ソフト

Solaris へサードパーティ製ソフトをどうやってインストールしているかというと、

- Sun 謹製 pkg（Companion Software <http://wwws.sun.com/software/solaris/freeware/> 含む）
- NSUG CD/DVD の pkg か、NSUG Packages Download Site <http://dld.nsug.or.jp/xoops/modules/mydownloads/> の pkg
- SunSITE Japan <http://sunsite.sut.ac.jp/sun/solbin/> でミラーされている sunfreeware.com <http://www.sunfreeware.com/> 提供の pkg
- SunSITE Japan <http://sunsite.sut.ac.jp/sun/solaris-binaries/{sparc,i86pc}/> でミラーされている SOLARIS PACKAGE ARCHIVE <http://ibiblio.org/pub/packages/solaris/{sparc,i86pc}/> 提供の pkg
- その他の個人・組織が提供する pkg
- 自分で make してインストール

といったところでしょうか。

しかし、上二つは、ソフトが古すぎたり、欲しいソフトがなかったりして、使えなかったりします。（最近はそんなことも少ないかもしれませんが。）また、まんなか三つも、なかなかほしいソフトがなかったりして、困ります。

結局、一番下の方法をとることになります。しかし、更新管理がめんどくさく、なんらかのパッケージ管理が欲しいところです。FreeBSD でいう、ports のようなしくみがあったら、楽だなぁ、なんて。

NetBSD のパッケージシステムをつかおう

そこで、NetBSD のパッケージシステム (The NetBSD Package System) を使用することにしました。略称として "pkgsrc" と書きます。（そして、一般にパッケージソースと読まれます。）

ちなみに、FreeBSD/OpenBSD でいう ports と同じものです。（ちなみに、FreeBSD/OpenBSD でいう package(s) は、NetBSD では、コンパイル済 / バイナリーパッケージ (precompiled/binary package(s)) といいます。NetBSD で package といえば、pkgsrc に含まれる、個々のソフトのことです。）

pkgsrc は、もともと NetBSD 用のパッケージシステムとして、FreeBSD の ports をお手本に開発されましたが、今では、NetBSD に限らず、他のプラットホームでも利用できるようになりました。詳しくは

- http://www.netbsd.org/docs/software/packages.html
- The pkgsrc guide

を見てください。

初期設定について

上記 URI をみれば書くことはないのですが、一応書きます。次のように設定を行ないました。

bootstrap のインストール

まず、bootstrap-pkgsrc をとってきます。

2004/3/12 から、bootstrap-pkgsrc は、pkgsrc 内に取り込まれました。よって、pkgsrc の最新をとってくることになります。

■インストール準備

最低、gcc と GNU zip(gzip)、できれば GNU tar がいりますので、事前に準備しておいて下さい。Solaris 8 のインストールメディア（CD-ROM）のから、

- SOLARIS 8 SOFTWARE 2 OF 2：SUNWgzip（とあとで使う SUNWgpch）
- Software Companion：SFWgcc と SFWgcmn

を入れるか、なければ適当なところ (SOLARIS PACKAGE ARCHIVE がオススメ) からとってきて、インストールしておいて下さい。

■ bootstrap-pkgsrc をとってくる

pkgsrc の最新は、annonymous cvs で提供されていますが、定期的に tar ball でスナップショットが提供されています。

```
ftp://ftp.jp.netbsd.org/pub/NetBSD-current/tar_files/pkgsrc.tar.gz
```

or

```
ftp://ftp.netbsd.org/pub/NetBSD-current/tar_files/pkgsrc.tar.gz
```

これを、とってきて、適当なところに展開します。GNU tar と gzip がパスに通っていると、

```
tar xzfp pkgsrc.tar.gz
```

とすれば、カレントディレクトリに pkgsrc ディレクトリができていると思います。GNU tar がない場合は、gunzip と untar を別々にやればよいでしょう。bootstrap ディレクトリに移動します。

```
cd pkgsrc/bootstrap
```

次に、README.Solaris を読み、必要最低限のソラリスパッケージがインストールされているかどうか確認して下さい。

とってきたものを、root で実行します。

```
./bootstrap --prefix=/opt/bsd --pkgdbdir=/var/opt/pkgsrc/db --sysconfdir=/opt/bsd/etc
--varbase=/var/opt/pkgsrc
```

gcc にパスが通っていなければ、パスを通すか、CC 変数を設定して実行すればよいです。

```
env CC=/opt/gnu/bin/gcc ./bootstrap --prefix=/opt/bsd --pkgdbdir=/var/opt/pkgsrc/db
--sysconfdir=/opt/bsd/etc --varbase=/var/opt/pkgsrc
```

インストール位置 (LOCALBASE) が /opt/bsd、パッケージのインストール情報や依存関係などのデータベース (PKG_DBDIR) が /var/opt/pkgsrc/db、通常 /var 以下にインストールされるもの (VARBASE) は /var/opt/pkgsrc、各種パッケージの設定ファイルの位置は /opt/bsd/etc にしました。これらの位置をどこにすべきかは悩みどころですが、一応こうしました。

この引数で、bootstrap を実行すると、初期設定と、最低限のツールがインストールされます。

mk.conf の設定

次に、/etc/mk.conf を設定します。これは、pkgsrc の各種設定を書くものです。まず、pkgsrc/bootstrap/work/mk.conf.example があるので、これを /opt/bsd/etc/mk.conf にコピーします。

次に、設定を加えます。まず、実際にとってきたソースファイル (distfiles) のおきばしょは、デフォルトでは /opt/bsd/pkgsrc/distfiles です。しかし、pkgsrc 全体を削除して更新したりしたいときに、一々 distfiles を退避するのはめんどうです。そこで、おき場所を変更します。

```
DISTDIR=/opt/bsd/distfiles
```

としました。mkdir して、dir をつくっておきます。

同様に、バイナリーパッケージの置き場所も変更します。

```
PACKAGES=/opt/bsd/packages
```

とします。mkdir して、dir をつくっておきます。

gcc ほかの準備

最低、ちゃんと動く gcc・GNU patch があるようにして下さい。ちゃんと動くのがなければ、Solaris 8 の Companion CD か、適当なところ (SOLARIS PACKAGE ARCHIVE がオススメ) から、gcc・GNU patch をとってきて、いれておいて下さい。

サーチパスなどの設定

- /opt/bsd/bin と /opt/bsd/sbin にパスを通しておいて下さい。
- /usr/usb をサーチパスから削除して下さい。無用のトラブルのもとです。
- gcc・GNU patch にパスを通して下さい。なんらかの理由で、パスを通せないのなら、/etc/mk.conf に CC=/opt/gnu/bin/gcc とか、TOOLS_PLATFORM.patch=/opt/gnu/bin/patch などと書いてください。(これは後で、pkgsrc から同じものをインストールできたら、mk.conf から消しましょう。)
- 作業ユーザの locale に関する環境変数が、各種コマンドの出力に影響するので、作業ユーザの LANG を C に設定することをお勧めします。(こうすれば一番簡単です。LC_MESSAGES を C にするのでもいいようです。)

わたしは csh ユーザなので、次のように ~/.csrhc を設定しました。(source ~/.cshrc も忘れずに。:-)

```
                                        setenv                                                  PATH
"/opt/bsd/bin:/opt/bsd/sbin:/usr/local/bin:/bin:/usr/ccs/bin:/usr/ccs/lib:/etc:/usr/etc:/usr/openwin/bin:/usr/local/etc
  setenv MANPATH "/opt/bsd/man:/usr/local/man:/usr/man"
  setenv LANG C
```

pkgsrc の移動

ついでに、とってきたソースがもったいないので、pkgsrc を移動させましょう。

```
mv pkgsrc /opt/bsd
```

パーミッションの変更（オプション）

わたしは、あとで一般ユーザで作業がしたかったので、展開後、パーミッションをいじりました。

```
chmod -R g+w /opt/bsd/pkgsrc /opt/bsd/packages /opt/bsd/distfiles
chgrp -R manager /opt/bsd/pkgsrc /opt/bsd/packages /opt/bsd/distfiles
```

manager グループに入っている人は、make できるというわけです。

はじめにインストールするもの

GNU patch

やっとこれで各種ソフトをインストールできそうです。最初にやることは、GNU patch をいれます。（以下は、/opt/bsd/pkgsrc/ カテゴリー / ソフト名 の場合、カテゴリー / ソフト名と書きます。）

```
cd /opt/bsd/pkgsrc/devel/patch
bmake
bmake install
bmake clean clean-depends
```

で OK です。（FreeBSD ports の場合は、make clean で依存先 dir もきれいにしてくれますが、pkgsrc の場合はそうではありません。あと、bmake clean と bmake clean-depends は必ず実行して下さい。無用のトラブルを回避できます。）一般ユーザで、bmake install しても、su - root できるユーザであれば、途中で root パスワードを聞いてくるので、OK です。

GNU make

次に、devel/gmake をいれます。

```
cd /opt/bsd/pkgsrc/devel/gmake
bmake
bmake install
bmake clean clean-depends
```

gcc

そして、最後に lang/gcc を入れます。

```
cd /opt/bsd/pkgsrc/lang/gcc
bmake
bmake install
bmake clean clean-depends
```

インストール直後のメッセージに出たように、/etc/mk.conf に一行追加しておきます。

```
.include "/opt/bsd/share/examples/gcc-2.95.3/mk.conf"
```

lang/gcc は gcc 2.95.3 なので、gcc3 がいいよーという人は、lang/gcc3-{c,c++,f77} を同様にいれて下さい。

ここまでインストールしたパッケージ一覧

下記のとおりにならなくてもかまいません。（Ver. 番号は違うと思います。）

```
% pkg_info -a
bootstrap-mk-files-20061111 *.mk files for the bootstrap bmake utility
nawk-20050424       Brian Kernighan's pattern-directed scanning and processing l
nbsed-20040821      NetBSD-current's sed(1)
tnftp-20050625      The enhanced FTP client in NetBSD
mtree-20040722      Utility for mapping and checking directory hierarchies
pax-20060202        POSIX standard archiver with many extensions
pkg_install-20061103 Package management and administration tools for pkgsrc
pkgmanpages-20050911 Manual page(s) for the packages collection
digest-20060826     Message digest wrapper utility
patch-2.5.4nb2      Patch files using diff output
libtool-base-1.5.22nb4 Generic shared library support script (the script itself)
pkg_install-info-4.5nb3 Standalone GNU info file installation utility
libiconv-1.10nb3    Character set conversion library
gettext-lib-0.14.6  Internationalized Message Handling Library (libintl)
gettext-tools-0.14.6 Tools for providing messages in different languages
gmake-3.81          GNU version of 'make' utility
gcc-2.95.3nb7       GNU Compiler Collection, version 2
```

あとかたづけ

一番最初にインストールした（pkgsrc からインストールしていない方の）、gcc や GNU patch をパスから外します。/etc/mk.conf に書いていたひとは、消してください。必要に応じて、アンインストールしてもいいでしょう。

libtool-base のいれなおし

このあと、最初に入れた gcc をアンインストールするならば、libtool-base をいれなおした方がよいです。（たとえば、database/db4 などを make すると、オブジェクトファイルをつなぐとき、失敗します。）

次に、pkgtools/pkg_tarup をインストールします。

```
cd /opt/bsd/pkgsrc/pkgtools/pkg_tarup
bmake
bmake install
bmake clean clean-depends
```

次に、libtool-base をいれなおします。replace という make ターゲットを使います。

```
cd /opt/bsd/pkgsrc/devel/libtool-base
bmake
bmake replace
bmake clean clean-depends
```

おすすめのパッケージ

あとは必要なソフトをいれていって下さい。ここからは、root で作業しなくても、大丈夫でしょう。無用のトラブルを避けるためにも、コンパイルユーザのサーチパスに、/usr/ucb や他の pkgsrc 以外の gcc を含んだパス（/usr/local/bin とか /opt/gnu/bin とか）が、/opt/bsd/{bin,sbin} より前にこないようにしてください。（できればサーチパスからはずすとよいと思います。特に、/usr/ucb。）

オススメは、まずは、

- archives/gtar-base
- archives/gzip-base
- misc/ja-less or misc/lv
- sysutil/findutils
- devel/diffutils
- sysutil/coreutils
- textproc/grep
- textproc/gsed

などでしょうか。

pkgsrc の更新

FreeBSD の ports では、ports 自体 cvsup によって、アップデートしていますね。Solaris 上の pkgsrc でも同じようなことをしようという話です。

前準備

pkgsrc のアップデートの方法は、

- いちいち ftp で tar ball をとってくる。
- cvsup を使う。
- sup を使う。
- anonymous cvs を使う。
- rsync を使う。

という方法が NetBSD Project によって提供されています。cvsup は i386 アーキテクチャきめうちなので、sparc アーキテクチャ上では使えません。sup は NetBSD 以外では使われてない方法ですし、OpenBSD プロジェクトで採用されなかったのは、信頼性にかけるからしいです。（わたしもうまくいったためしがありません。）ゆえに、anonymous cvs か rsync かになります。rsync はやたら負荷が高いような感じなので、ここでは、anonymous cvs を選択しましょう。

まず、devel/cvs と security/openssh をインストールします。もし、cvs がインストールできず、anonymous cvs を使いたい場合は、cvs だけ、Solaris Package Archive の cvs を使うという手もあります。ここでは、cvs を用意できたとして、先にすすみます。あと、古い distfiles の掃除のために、pkgtools/pkglint もインストールしておいて下さい。

cvs update の時に、*.html のエラーが大量に出るので、うっとうしければ、事前に消しておいてもいいでしょう。

```
cd /opt/bsd/pkgsrc
find . -name "*.html" -print | xargs rm -f
```

手動で行う場合

まず、pkgsrc ソースツリーをきれいにします。確実にするには、root でするといいですね。（わたしはいつも sudo を使っています。）

```
cd /opt/bsd/pkgsrc
bmake clean
```

ものすごく時間がかかるので、お茶でも一杯。

そんなに待てない、という人は、pkgtools/pkgclean をインストールして、使ってみてもいいと思います。超高速です。（でも、Ver. によっては、solaris では make できなくなることがあります。）

次に、cvs update します。一般ユーザが pkgsrc の更新ができない場合は、root でしましょう。

```
setenv CVS_RSH ssh
cvs update -dP
```

これで pkgsrc ツリーが更新されました。

次に、古い distfiles を削除するために、lintpkgsrc を使います。

```
lintpkgsrc -ro
```

最後に、一般ユーザ権限で実行できるようにしている場合は、パーミッションをもとにもどします。

```
cd /opt/bsd
chmod -R g+w pkgsrc pkgsrc_local_patch distfiles
chown -R root:manager pkgsrc pkgsrc_local_patch distfiles
```

これでおしまい。

自動化する場合

自動化が必要であれば、root の crontab に書いておけばいいですね。参考までに、アップデートのための sh スクリプトをあげておきます。

```
#!/usr/xpg4/bin/sh

cd /opt/bsd/pkgsrc && \
/opt/bsd/bin/bmake clean >/dev/null 2>&1 && \
env CVS_RSH=/opt/bsd/bin/ssh /opt/bsd/bin/cvs update -dP >/dev/null 2>&1 && \
/opt/bsd/bin/lintpkgsrc -r >/dev/nll 2>&1 && \
cd /opt/bsd && \
chmod -R g+w pkgsrc pkgsrc_local_patch distfiles >/dev/null 2>&1 && \
chgrp -R manager pkgsrc pkgsrc_local_patch distfiles >/dev/null 2>&1
```

次に、cron にこのように書きます。上記の内容を /opt/bsd/sbin/update_pkgsrc.sh に保存したとして、

```
0 3 * * 1       /opt/bsd/sbin/update_pkgsrc.sh >/dev/null 2>&1
```

これで、更新が自動化されました。

セキュリティアラートの自動化

pkgsrc には、インストールしたパッケージに関する脆弱性情報が公開されていないかを調べるコマンドがあります。bootstrap をインストールした時にいっしょにインストールされているはずで

す。定期的に実行して、結果をメールで受け取るようにしましょう。

・/opt/bsd/sbin/download-vulnerability-list：脆弱性情報データをとってくる
・/opt/bsd/sbin/audit-packages：脆弱性情報があるかどうか調べる

あとは、cron で実行して、メールで送るようにすればいいだけですね。

参考までに、crontab はこんな感じ。

```
0   9   *   *   1             /opt/bsd/sbin/download-vulnerability-list   >/dev/null   2>&1   &&
/opt/bsd/sbin/audit-packages   |   /usr/bin/mailx   -s   'hoge.example.jp   pkgsrc(/opt/bsd/..)   weekly
insecurity output' root@hoge.example.jp
```

これでおしまい。

pkg_info pkg_install して、MESSAGE を読むともう少し詳しいことが書いてあります。

パッケージのアップデート

パッケージをアップデートする時の注意など。

pkgsrc には、FreeBSD でいう portupgrade はありません。したがって、update したいパッケージが、たとえば shell/zsh なら、

```
cd /opt/bsd/pkgsrc/shell/zsh
bmake update
```

とをすればいいことになります。

... しかし、仮に bmake がとおらないとなると、パッケージはアンインストールされたわ、bmake はとおらないわ、で泣きを見ることになります。そこで、下記のようにすることを推奨します。

まず、現 Ver. の、バイナリーパッケージがあるかどうかを確認します。/opt/bsd/packages 以下にあるはずです。なければ、万が一にそなえて、つくっておきます。pkgtools/pkg_tarup をインストールすれば、

```
pkg_tarup パッケージ名
```

で今インストールされているものからバイナリーパッケージをこしらえてくれます。これを、一時的に /tmp とかにとっときます。そして、

・ bmake
・ bmake deinstall
・ bmake package

とします。最初の "make" で、make がとおるか確認できます。make がとおれば、だいたい OK ですから、"make deinstall" で現行 Ver. をアンインストールして、次に、新 Ver. をインストールします。"make package" でインストールすると、同時にバイナリーパッケージをつくってくれるので、

それでいれます。

もし、インストールの途中に失敗したとか、make がとおらなかったとか、そういったときは、とってあったバイナリーパッケージを pkg_add すればもとにもどせます。

依存関係がある時は、make deinstall のところで、依存関係により、アンインストールできない旨警告されます。そのときは、依存先をひとつづつアップグレードしていくのが一番確実です。

libiconv のような、たくさんのパッケージに関係しているパッケージは、セキュリティアップデートに関係しない限り、手を出さないほうがよいです。

たんじゅんなパッケージ（ただのスクリプトとか）は、上書きアップグレードしてもいいかもしれません。上書きインストールする時は、

```
bmake replace
```

です。ただし、原則として使わない方がいいと思います。

bmake package を使うようになると、忘れるのが、clean と clean-depends です。bmake package したあとは、忘れずに。

FAQ

bootstrap 自体が make できません。

そういうこともあります。bootstrap 自体の更新をまって、再度 cvs してみるか、http://www.netbsd.org/Documentation/software/packages.html でバイナリが提供されているので、それをつかってみてもいいでしょう。（ただし、インストール位置は自由に選べません。）

いしはらのささやかな経験では、自分で make した gcc だとうまくいかないことがありました。すなお？に、どこかのパッケージ化された gcc を使ってみましょう。

最初に lang/gcc や lang/gcc3-c を make しようとしたら、libgcc_s.so.1 が見つからないと言われる。

とりあえず、sunfreeware.com とかから、gcc3 をとってきてインストールして、さっそく pkgsrc の gcc を make しようとすると、

```
===> Building for gcc-2.95.3nb7
ld.so.1: gmake: fatal: libgcc_s.so.1: open failed: No such file or directory
Killed
*** Error code 137
```

などといわれたりします。

これは、LD_LIBRARY_PATH を libgcc_s.so.1 のある dir に設定してやると make できます。/usr/local/lib がそうだとすると、

```
env LD_LIBRARY_PATH=/usr/lib:/usr/local/lib bmake
```

で make できるはずです。

crle（solaris の ldconfig）で、ライブラリパスをホストに設定することも可能ですが、上記で対応できますし、トラブルの元なので、おすすめしません。

LD_LIBRARY_PATH？ crle？っていうひとは、Solaris.RougeLife.org さんのページを見ましょう。

devel/gmake をインストールしましたが、動きません。

lang/gcc3 を gcc に使っている環境だと、

```
ld.so.1: gmake: fatal: libgcc_s.so.1: open failed: No such file or directory
```

Killed となるかもしれません。

ldd すると、

```
# ldd /opt/bsd/bin/gmake
      libkstat.so.1 =>         /usr/lib/libkstat.so.1
      libintl.so.3 =>  /opt/pkg/lib/libintl.so.3
      libiconv.so.2 =>         /opt/pkg/lib/libiconv.so.2
      libc.so.1 =>     /usr/lib/libc.so.1
      librt.so.1 =>    /usr/lib/librt.so.1
      libgcc_s.so.1 =>         (file not found)
      libgcc_s.so.1 =>         (file not found)
      libdl.so.1 =>    /usr/lib/libdl.so.1
      libaio.so.1 =>   /usr/lib/libaio.so.1
      /usr/platform/SUNW,Sun-Fire-280R/lib/libc_psr.so.1
```

libgcc_s.so.1 がありません。

でも、

```
  # dump -Lv /opt/bsd/bin/gmake | grep PATH
    [8 ]             RUNPATH                          /opt/bsd/lib:/opt/pkg/gcc3 /lib:/opt/bsd/gcc3
/lib/gcc-lib/sparc-sun-solaris2/3.3.6
  [9]     RPATH            /opt/bsd/lib:/opt/pkg/gcc3/lib:/opt/bsd/gcc3/lib/gcc-lib/sparc-sun-solaris2
/3.3.6
```

と RPATH があって、

```
  > find /opt/bsd/lib /opt/bsd/gcc3/lib    /opt/bsd/gcc3/lib/gcc-lib/sparc-sun-solaris2/3.3.6 | grep
libgcc_s.so.1
  /opt/bsd/gcc3/lib/libgcc_s.so.1
```

と、RPATH 上にあったります。

ktruss すると、/opt/bsd/lib までしか探してくれないみたいです。ktruss の最後は

```
close(3)                                        = 0
stat("/opt/bsd/lib/libgcc_s.so.1", 0xFFBEF660)  Err#2 ENOENT
stat("/usr/lib/libgcc_s.so.1", 0xFFBEF660)      Err#2 ENOENT
write(2, " l d . s o . 1 :   g m a".., 77)      = 77
munmap(0xFF390000, 8192)                        = 0
lwp_self()                                      = 1
        *** process killed ***
```

でした。うーん。

これ、なんでおきるかわからないのですが、/opt/bsd/lib に libgcc_s.so.1 をシンボリックリンクするととりあえず使えるので、そうしてください。あるいは、/opt/bsd/gcc3/lib を LD_LIBRARY_PATH に追加しましょう。

make できません。

まず、第一に考えられることは、

- Solaris デフォルト +/opt/bsd/{bin,sbin} 以外がサーチパスに含まれている
- bmake(BSD make) ではなく GNU make をつかった
- ftp でとってきたファイルが壊れている

でしょうか。これを確認して下さい。

それでも問題なければ、make できないんだと思います。pkgsrc に含まれるすべてのアプリケーションが solaris os で make できるわけではありません。

devel/cvs をインストールして、anonymous CVS を試したのですが、pkgsrc のアップデートができません。

すでに pkgsrc 以外でインストールしているなどの理由で、security/openssh をインストールしていないのでしょう。こういうときは、

```
env CVS_RSH=/path/to/ssh cvs update -dP
```

などと、CVS_RSH 変数に ssh のパスを入れてください。

FreeBSD の portupgrade に含まれる portversion みたいなのはないのか。

pkgtools/pkglint に含まれる、lintpkgsrc で、

```
lintpkgsrc -i
```

で OK です。

pkgtools/pkg_chk で、

```
pkg_chk -i
```

でも OK。こちらの方が速いみたい。

使い方がいまいちよくわからない。

NetBSD のパッケージシステムとほぼ同じ操作が可能です。NetBSD Project の Web や、pkgsrc/doc/pkgsrc.{txt,html} も参照して下さい。

入っているソフトの一覧がほしい。

pkg_info -a で OK です。pkg_info の man も参考にしてください。

パッチあてに失敗する。パッチファイルは壊れていないし、linux や NetBSD/i386 上でやると正常にあてられる。

オリジナルファイルの最後に newline がなくて、最後の行の次の行にパッチがあたると、おかくしなるようです。

```
env TOOLS_PLATFORM.patch='/opt/bsd/bin/gpatch -b -l' bmake
```

を試してみてください。('-l' がポイントです。)

/bin/sh の引数のスクリプトのところで make が止まるようだ。

/bin/ksh か、/usr/xpg4/bin/sh ならばうまくいくはずです。

```
env TOOLS_PLATFORM.sh=/usr/xpg4/bin/sh bmake
```

などとするか、一時的に入れ替えるかなにかしてしのいでください。だめなら、shells/bash をいれて、それと一時的に入れ替えるか何かしてしのいでください。(こっちのほうが確実。)

install のところで、install コマンドの引数がおかしいようで止まる。

/usr/ucb/install を呼び出しているかどうか確認してください。/usr/sbin/install じゃだめです。/usr/ucb/install でもだめなら、sysutils/coreutils をインストールして、

```
env TOOLS_PLATFORM.install=/opt/bsd/bin/ginstall bmake
```

などとするか、一時的に /usr/ucb/install と置き換えてください。

/usr/ucb/install は引数を 2 つとることができないみたいなので、原因が引数を 2 つとっていることなのであれば、そこだけ Makefile を書き換えてもいいかもしれません。

awk を実行するところで、エラーが多発する。

これは、そのパッケージが gawk を前提にしているからでしょう。しかし、solaris の場合は、OS に付属している nawk を使うのがデフォルト設定です。ですから、lang/gawk をいれて、

```
.if exists(${LOCALBASE}/bin/gawk)
TOOLS_PLATFORM.awk=/opt/bsd/bin/gawk
.endif
```

などと /etc/mk.conf 書いておけば、そうなります。

なお、grep や sed や xargs でも同じことがいえるので、同様に pkgsrc から GNU のものをインストールして、対策して下さい。

日本語のとおる vi はどこに？

あなたは OS 付属の vi を忘れています。

```
env LANG=ja /usr/xpg4/bin/vi
```

を試してみましょう。

deve/gmake や archives/gtar-base などをインストールするときに、*.po ファイルのインストールのところで、エラーが出る。

環境変数 LANG を C にして再度試してみて下さい。（mk/subst.mk の中で、file コマンドの出力を判断して動作が変わるところがあるのですが、file コマンドの出力が日本語だとおかしくなるのです。）

```
bmake clean clean-depends
env LANG=C bmake
...
```

同じ原因で、x11/kterm も install の途中でおかしくなるようです。

また、同様の原因で、lang/perl58 でも、/bin/sh の実行のところで、ulimit の値が正しくないなどといわれて止まるみたいです。（これは、mk/platform/SunOS.mk の中で、ulimit コマンドの出力を判断して動作が変わるところがあるのですが、ulimit コマンドの出力が日本語だとおかしくなるのです。）

ただし、2005/5/20 頃、mk/subst.mk の修正がされたようで、もう上記の問題はおきないと思います。

バイナリーパッケージは提供されていないの？

http://www.netbsd.org/Documentation/software/packages.html をみてみてください。Solaris 9/sparc はあるようです。（ただし、インストール位置は変更できません。）

root ユーザ以外で、bmake install すると、root パスワードを要求されるのはいいんだけど、ここに sudo を使えないかな？

pkgsrc.txt に書いてあります。

```
.if exists(${LOCALBASE}/bin/sudo)
SU_CMD=${LOCALBASE}/bin/sudo /bin/sh -c
.endif
```

を、/etc/mk.conf に書きましょう。（sudo をインストールしていない場合は、security/sudo を先にインストールして、必要な設定をしてからです。）

ruby 関連のパッケージなどでは、リンカが失敗する。

[netbsd,08980] によれば、Solaris 8 はそうみたいです。作業ユーザに LD_LIBRARY_PATH を設定しましょう。

```
cd /usr/pkgsrc/lang/ruby18-base
setenv LD_LIBRARY_PATH "/opt/bsd/lib:/lib:/usr/lib:/usr/local/lib:/usr/dt/lib:/usr/openwin/lib"
bmake
bmake package
bmake clean clean-depends
unsetenv LD_LIBRARY_PATH
```

とか。

ただ、もうメンテナーの方がこの問題は直されていると思います。

GNU アプリケーションをインストールすると、コマンドの頭に g がつくのがうっとおしい。なんとかならん？

GNU_PROGRAM_PREFIX という変数が頭につく「何か」ですので、これを空にしましょう。

```
cd /usr/pkgsrc/textproc/gsed
env GNU_PROGRAM_PREFIX='' bmake
env GNU_PROGRAM_PREFIX='' bmake package
env GNU_PROGRAM_PREFIX='' bmake clean clean-depends
```

などとすれば、gsed ではなく、sed でインストールされます。

ただし、一部のコマンドは、そうなりません。あと、devel/gmake はそのままインストールしたほうがいいです。

依存関係で xxx を要求される。できれば、すでに別にインストールしてある xxx を使いたい。

TOOLS_PLATFORM.xxx という変数を設定すれば、そのツールは依存関係があっても、インストールされず、その変数に指定したコマンドが用いられます。

```
env TOOLS_PLATFORM.perl=/path/to/perl bmake
```

などとすればいいですね。ただし、xxx はなんでもいいわけではありません。xxx に何が書けるのかは、mk/tools 以下の defaults.mk をのぞいてみてください。ただし、tools.SunOS.mk で指定されているものは、最初から Solaris にあるコマンドを使うようになっていますので、改めて指定する必要はありません。（tools.SunOS.mk の指定を上書きしたいときは、もちろん必要です。）

mk.conf に書いておくと、常に指定したコマンドが使われるようになります。

現在の設定がみたい場合は、make show-tools でみることができます。

gcc じゃなくて、Sun WorkShop を使いたいんだけど。

doc/pkgsrc.{txt,html} の、「3.3.7.2. If you are using Sun WorkShop」と「3.3.7.3. Buildling 64-bit binaries with SunPro」を見て下さい。bootstrap から使いたいのであれば、bootstrap/README.Solaris の該当項目を参照して下さい。

libtool が動くところで、/bin/ksh が segmentation fault することがある。

doc/pkgsrc.{txt,html} の「3.3.7.4. Common problems」にあるとおり、たとえば shells/bash などをインストールして、

```
.if exists(${LOCALBASE}/bin/bash)
CONFIG_SHELL=   ${LOCALBASE}/bin/bash
WRAPPER_SHELL=  ${LOCALBASE}/bin/bash
.endif
```

を /etc/mk.conf に書いて、devel/libtools-base をインストールしなおして下さい。

pkgtools/pkg_install を update しろ、と言われたけど、update できない。

security/audit-packages をインストールしていませんか。このパッケージは pkgtools/pkg_install に統合されたので、もうありません。先に uninstall しましょう。

```
pkg_delete audit-packages
```

pkgsrc にはほしいソフトがないんだけど。

pkgsrc-wip にはあるかもしれません。pkgsrc-wip は、pkgsrc に取り込まれる前の、作業場所のようなところです。使い方はリンク先を参照して下さい。

pkgsrc-wip-jp もあります。

pkgsrc にどんなソフトがあるのかわかりにくい。

online でよければ、http://pkgsrc.se/ があります。pkgsrc-wip のものまで探してしまいますが、探しやすいと思いますよ。

GUI のフロントエンドはないの？

Debian GNU/Linux でいう、dselect や tasksel のような、コンソール指向のものは、pkgtools/pkg_select があります。（ただし、いしはらは Solaris では使ったことがありません。）

Fedora Core でいう、gyum のような、X な GUI のものは、残念ですがありません。

Zoularisってなんですか？

bootstrap-pkgsrc ができる前の、同様のしくみです。bootstrap-pkgsrc にとって代わられ、既に使われていません。

Solaris 10 で試しているんですが、いまいちうまくいきません。

onetbsd.org で知ったのですが、Pkgsrc and Solaris10-amd64 を見てみてはどうでしょうか。Solaris 10 の amd64 と sparc64 で gcc な人の Howto みたいです。あと、一番下のりんく集もどうぞ。

何か情報源は？

tech-pkg-ja ML と、netbsd ML ぐらいしか思いつきません。試行錯誤して、困ったら、どちらかで聞いてみるといいでしょう。過去 log ものぞいてみるといいかもしれません。

英語でよければ、pkgsrc-users ML がオススメです。

pkgsrc はどうも自分にあわない。他にいいのはないの？

いっそのこと Solaris ではなく、NetBSD/sparc, NetBSD/sparc64, NetBSD/i386, NetBSD/amd64 てのはどう？

... じょうだんですよ、冗談。ええと、いろいろ有名なのはあるみたいなのですが、Community Open Source Software Distribution for The Solaris Operating Environment <http://www.blastwave.org/> や、OpenPKG <http://www.openpkg.org/> なんかどうでしょうか。

pkgsrc のように、ソースベースなものがいい、というのであれば、Gentoo Linux の Portage の Solaris 版である Portaris があります。（が、portaris.org はなくなってしまって、ドメイン取られてしまいましたね ...。いったいどこへいったんでしょうか。）

（続く）

りんく

少しだけ。

- NetBSD's pkgsrc on Solaris
- docs:solaris:pkgsrc(google キャッシュ)
- How to use pkgsrc on Solaris
- Bootstrapping pkgsrc on x86 Solaris 9