Design, Implementation and Operation of NetBSD Base System Packaging

Chitose Institute of Science and Technology

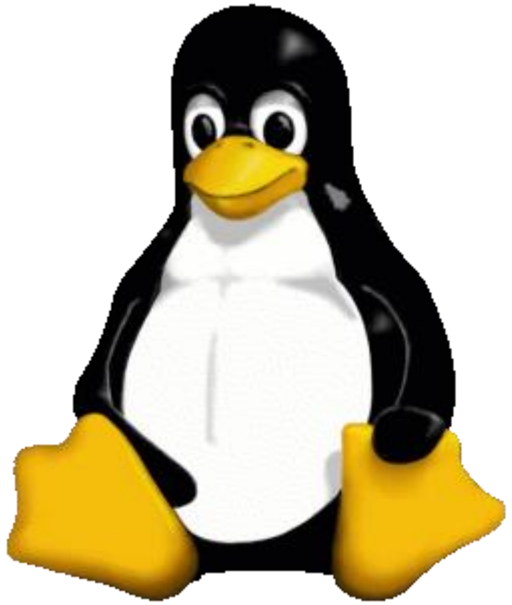
Yuuki Enomoto

Ken’ichi Fukamachi

Abstract

- OS built on fine granular small parts is preferable to one built on the large tarballs in order to ……
  - Speedy security update.
  - Easy replacement.
  - Rollback

- In Linux distributions, the system are already divided into many small packages.



Copyright (c) lewing@isc.tamu.edu
Larry Ewing and The GIMP

Abstract

- BSD Unix are behind the curve on the base system packaging.
- We have developed a software "**basepkg**" to improve NetBSD base system granularity.
- This presentation shows replacement of a few OS granular parts is clearly faster and can provide extra useful functions for NetBSD users.

Contents

Contents

Background (1/6)

- OS has been managed on
  one source tree set has been managed.

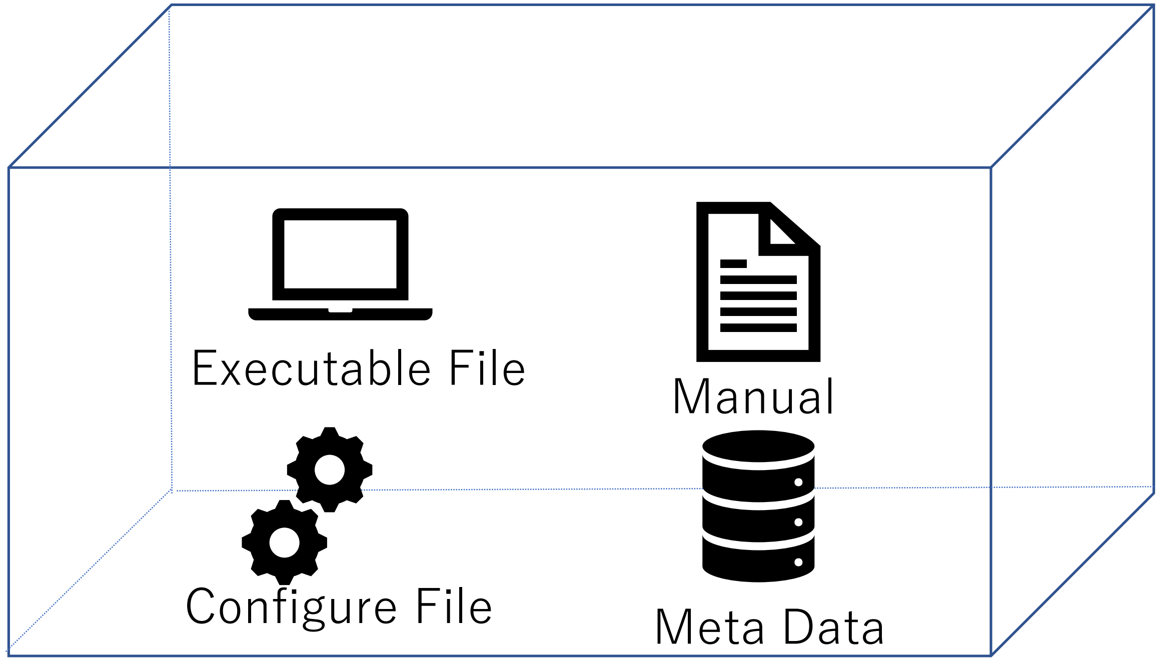
- In NetBSD, either of a large or small tarball or
  the combination is used.
  - **base.tgz** (Mandatory for the operating system)
  - **comp.tgz** (Compiler tools)
  - **man.tgz** (Manual)

```
/usr/src/
|-- BUILDING
|-- CVS
|-- Makefile
|-- Makefile.inc
|-- UPDATING
|-- bin
|-- build.sh
|-- common
|-- compat
|-- crypto
|-- dist
|-- distrib
|-- doc
|-- etc
|-- external
|-- extsrc
|-- games
|-- gnu
|-- include
|-- lib
|-- libexec
|-- regress
|-- rescue
|-- sbin
|-- share
|-- sys
|-- tests
|-- tools
|-- usr.bin
|-- usr.sbin
`-- x11
```

Background (2/6)

- Third-party software are managed as a set of small archives called as “package”.

- Package consists of software, documentation, configuration files, and meta data.

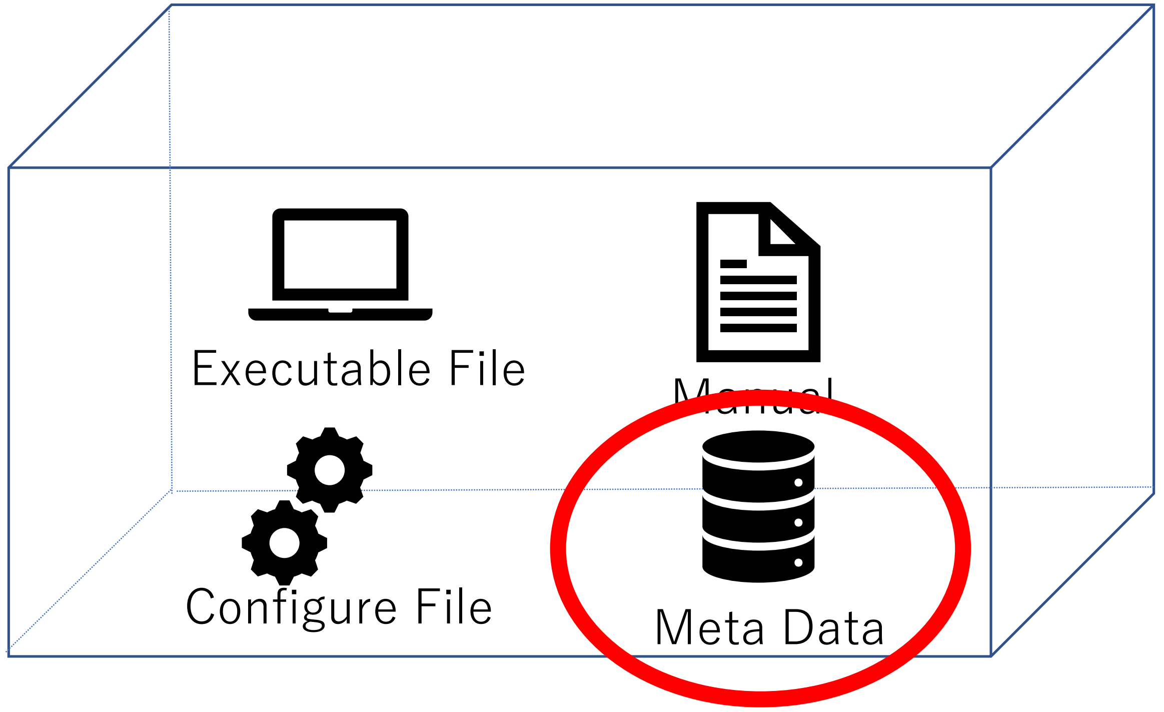


Background (3/6)

- Meta data required to operate in installation and deinstallation.

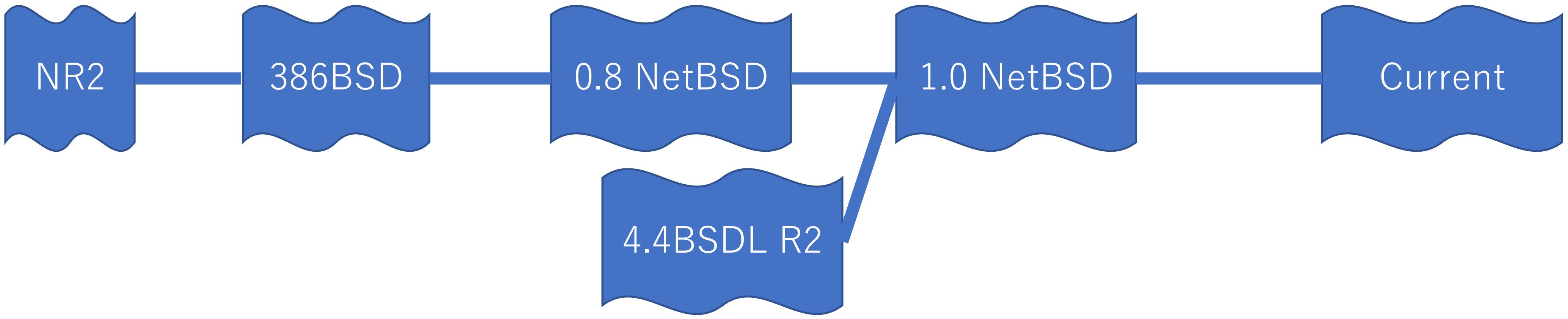
OS	format	package manager
FreeBSD	.txz	**pkg**(7)
NetBSD	.tgz	**pkg_install**

- Meta data contains ……
  - Build environment
  - Comment
  - Description
  - Dependency
  - Install script

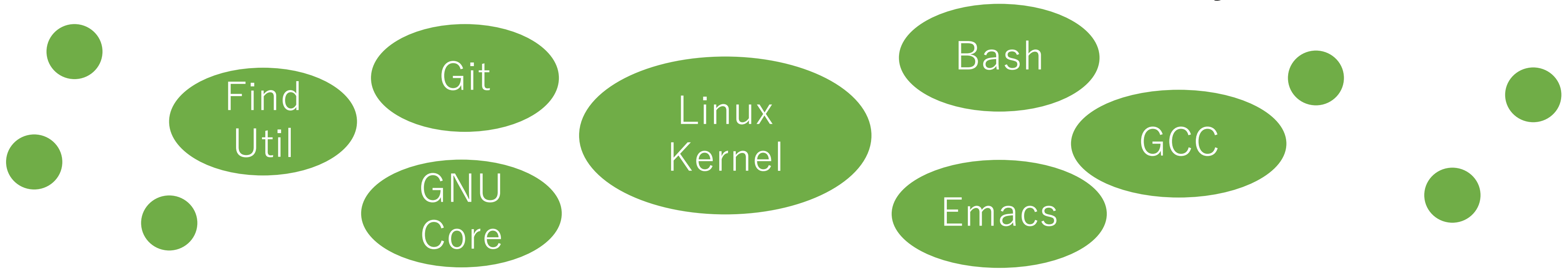


Background (4/6)

- Historically, BSD Unix has been developed in its own source tree including kernel and userland program.



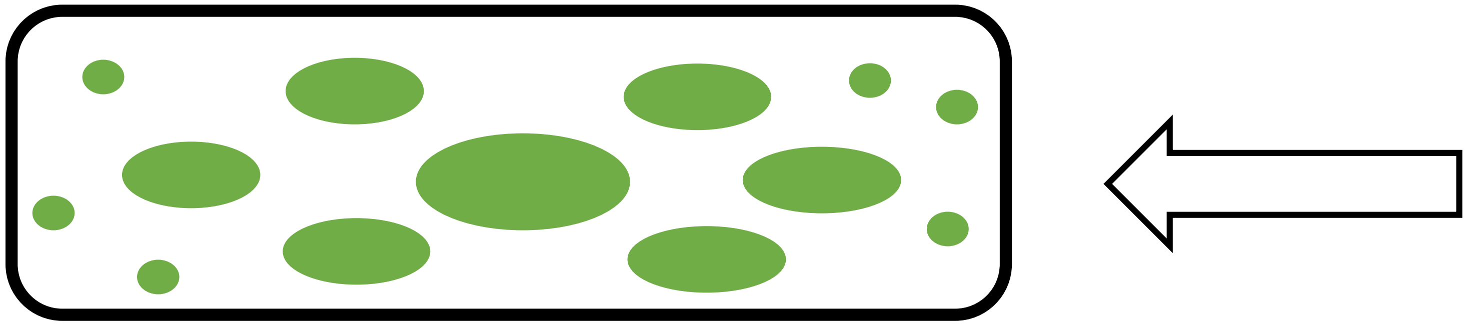
- Linux distribution needed to assemble a lot of system utilities.



Background (5/6)

- Major Linux distributions such as Debian and RHEL are already divided into many small packages.

- These OS's can manage both its own base system and third-party software through its package manager.



Managed by APT (Debian)

Background (6/6)

- BSD Unix have each package framework
  - **ports**(7)
  - **pkgsrc**(7)

  … but they have been used only
  for third-party software management.

- However today, for uses, it’s better that OS can be assembled on a lot of small parts.
  - It’s suitable for security update, replacement and rollback of specific parts.

Contents

Packages in BSD Unix (1/4)

- BSD Unix consists of the base system and optional third-party software not distributed within the base system.

- Third-party software provided by **ports**(7) or **pkgsrc**(7).

	ports(7)	**pkgsrc**(7)
Meta Data	**+COMPACT_MANIFEST** **+MANIFEST**	**+BUILD_INFO** **+CONTENTS** **+DESC** …
Package Manager	**pkg**(7)	**pkgtools/pkg_install**
Platform	FreeBSD, OpenBSD	Multi Platform

Packages in BSD Unix (2/4)

- BSD Unix has some approach for base system packaging.
- FreeBSD 11 introduced
  a base system packaging mechanism called “**PkgBase**”.
- NetBSD has a framework called “**syspkg**” introduced
  at January 8, 2002.
  - It’s also merged into `build.sh` as a feature
    of the official building process.
  - **`build.sh syspkgs`**

Packages in BSD Unix (3/4)

- Especially, in syspkg, NetBSD wiki says
  “There has been a lot of work in this area already,
  but it has not yet been finalized.”

- syspkg has several problems for these years.
  1. The database under **usr/src/distrib/sets** has been incomplete.
  2. Package made by syspkg lacks several meta data
     the current pkgsrc package.
  3. It’s possible to overwrite existing **/etc** files
     because it has not install script.

Packages in BSD Unix (4/4)

- We focused on NetBSD's **syspkg**.
  - Package is suitable for powerless architecture.
  - It look like there is room for advancement.

- It looks hard to directly fix **syspkg** framework which consists of a lot of makefiles, scripts, and undocumented data.

- We have developed another packaging mechanism as a third-party software by using only **syspkg** database and making the best use of **pkgsrc**(7) framework.

Contents

basepkg (1/12)

- We have developed a new framework “**basepkg**” that can packaging NetBSD base system instead of **syspkg**.
  - It distributed under the 2-cause BSD License.
  - It published on `https://github.com/user340/basepkg`

- Oct 26, 2016: Published on GitHub.
- May 19, 2017: Imported to **pkgsrc-wip**

- The latest version is 1.4.

basepkg (2/12)

- The feature comparison between **basepkg** and **syspkg** are shown at this table.

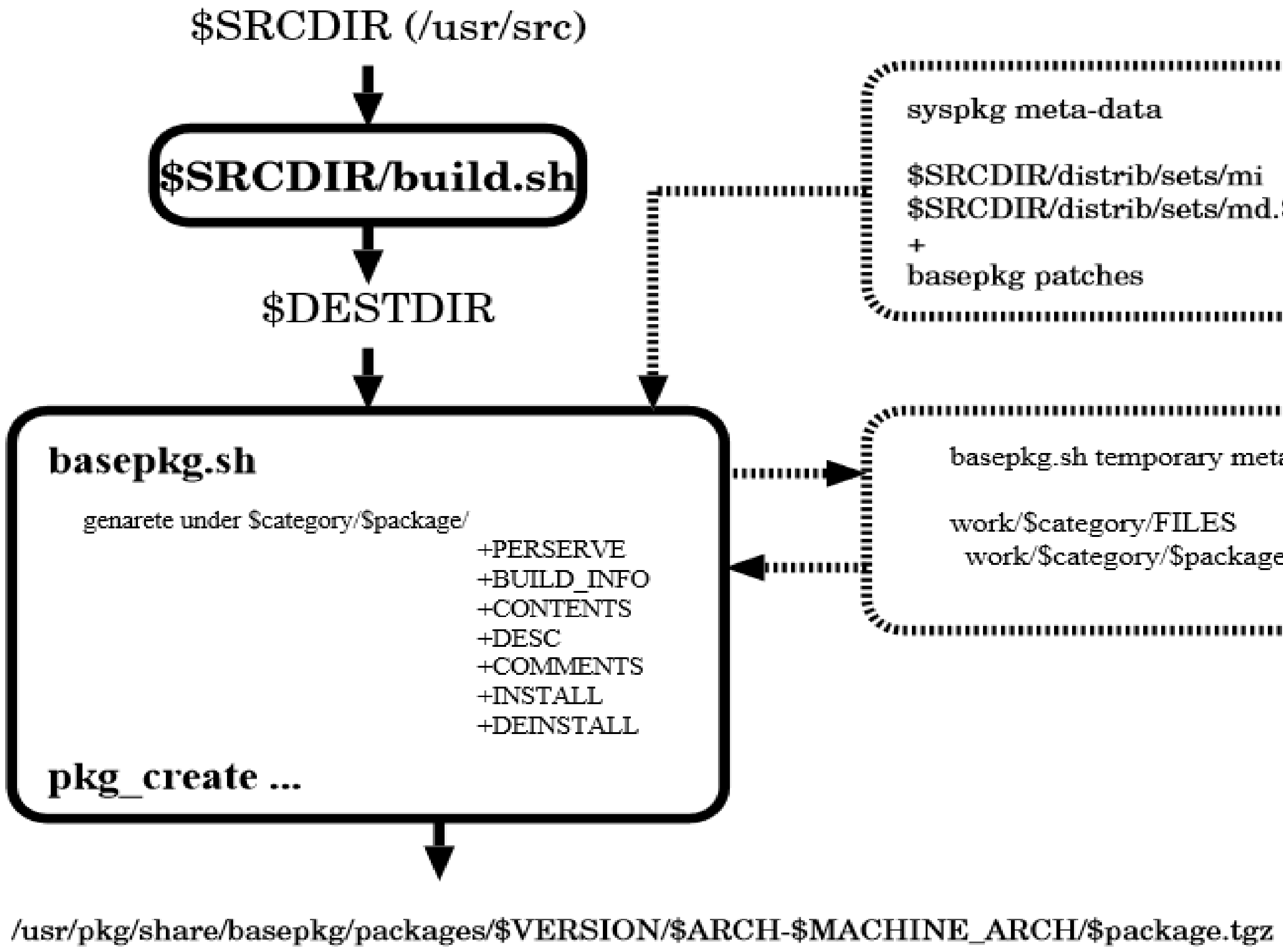
	syspkg	basepkg
Language	Makefile, Bourne shell	Bourne shell
Sum of Number of Lines	3,729 lines	1,190 lines
Execution Time	850.4 sec	1188.4 sec
Install Script	none	available
Kernel Package	none	available
Documentation, Report	none	ABC2017, 2018, ……
Develop Team	NetBSD Project	GitHub

basepkg (3/12)

- To write a sustainable program, **basepkg** is written to be POSIX compliant and portable as could as possible.
  - The latest code is POSIX compliant except …
    - **hostname**(1)
    - **mktemp**(1)
    - **pkg_create**(1)

- We use **ShellCheck** to validate and gain code quality and make the code warning-less as could as possible.
  - https://github.com/koalaman/shellcheck
  - https://www.shcellcheck.net

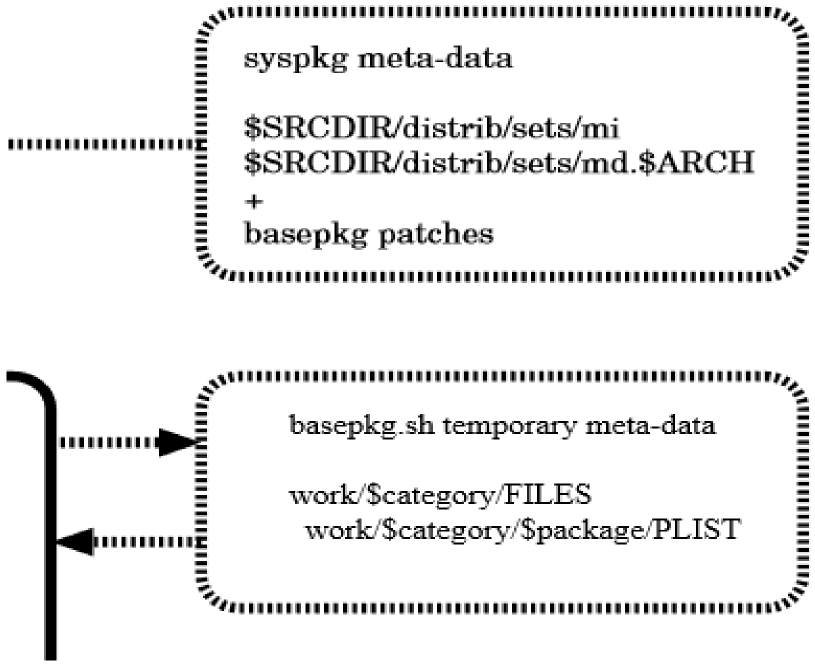
basepkg (4/12)



$SRCDIR (/usr/src)
$SRCDIR/build.sh
$DESTDIR
basepkg.sh
genarete under $category/$package/
+PERSERVE
+BUILD_INFO
+CONTENTS
+DESC
+COMMENTS
+INSTALL
+DEINSTALL
pkg_create ...
syspkg meta-data
$SRCDIR/distrib/sets/mi
$SRCDIR/distrib/sets/md.$ARCH
+
basepkg patches
basepkg.sh temporary meta-data
work/$category/FILES
work/$category/$package/PLIST
/usr/pkg/share/basepkg/packages/$VERSION/$ARCH-$MACHINE_ARCH/$package.tgz

basepkg (5/12)

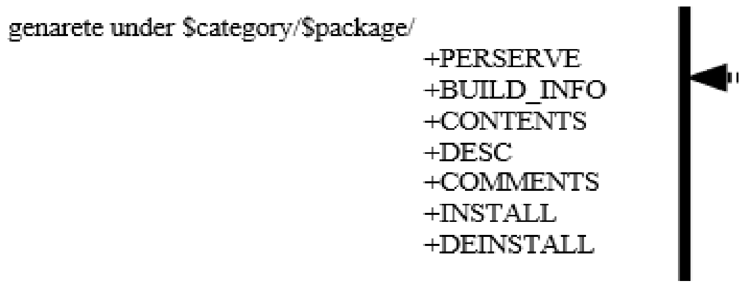
- basepkg reads list of a set of file name and package name from **`mi`** and **`md.ARCH`** under **`sets/lists`** directory.
- Then, basepkg generates temporary files.


syspkg meta-data
$SRCDIR/distrib/sets/mi
$SRCDIR/distrib/sets/md.$ARCH
+
basepkg patches
basepkg.sh temporary meta-data
work/$category/FILES
work/$category/$package/PLIST

basepkg (6/12)

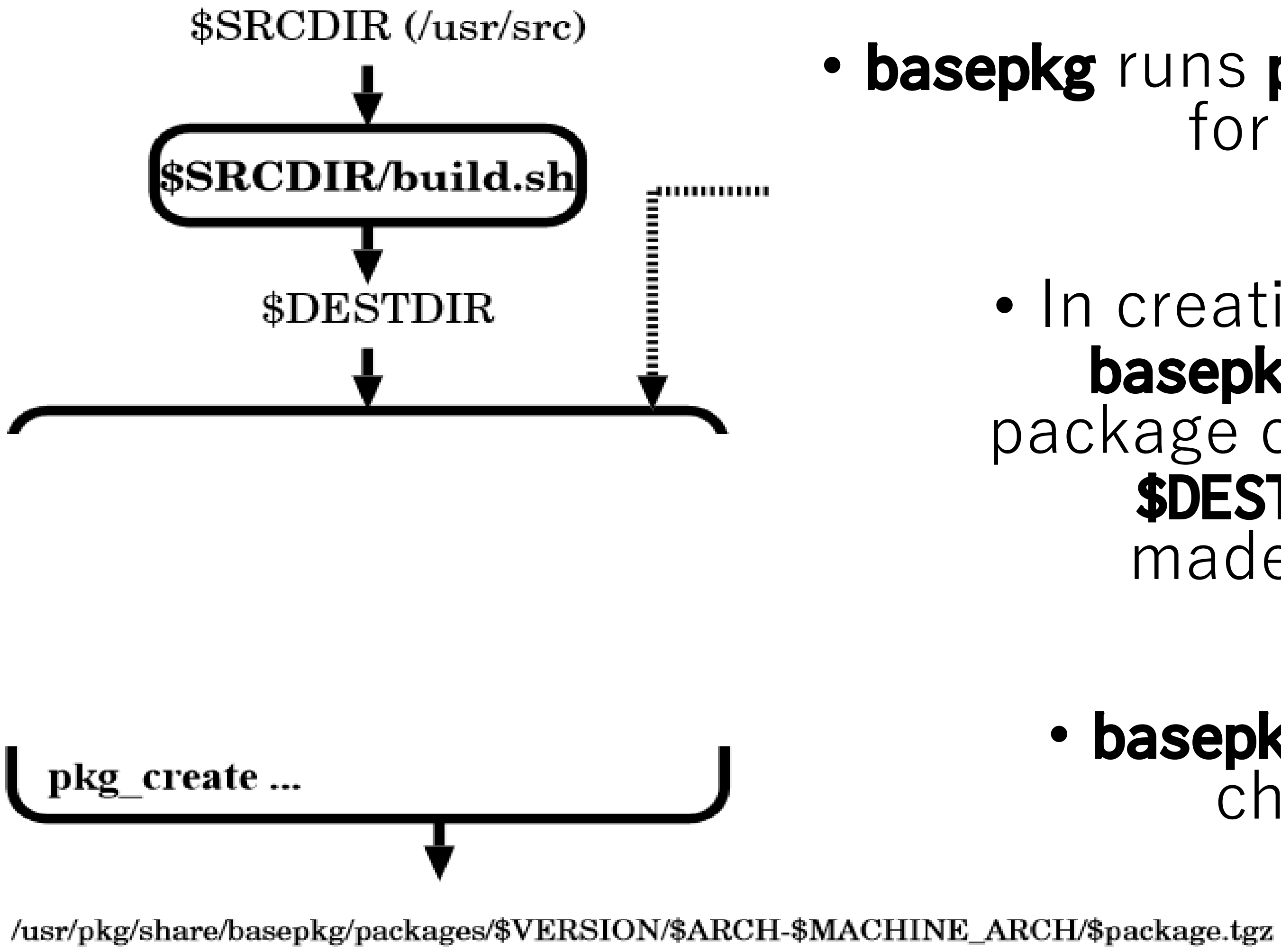
- **basepkg** emulates the generation of pkgsrc meta data.
- We supporting these meta data.

+BUILD_INFO
+COMMENT
+CONTENTS
+DEINSTALL
+DESC
+PRESERVE
+INSTALL
+SIZE_ALL
+SIZE_PKG
+PRESERVE



genarete under $category/$package/
+PERSERVE
+BUILD_INFO
+CONTENTS
+DESC
+COMMENTS
+INSTALL
+DEINSTALL

basepkg (7/12)


$SRCDIR (/usr/src)
$SRCDIR/build.sh
$DESTDIR
pkg_create ...
/usr/pkg/share/basepkg/packages/$VERSION/$ARCH-$MACHINE_ARCH/$package.tgz

- **basepkg** runs **pkg_create**(1) for all packages.
- In creating packages, **basepkg** gathers the package content under **$DESTDIR** directory made by **build.sh**.
- **basepkg** creates the checksum files.

basepkg (8/12)

- How to Install

```
# cd /usr/pkgsrc/wip/basepkg
# make install clean clean-depends
```

- How to build packages

```
# cd /usr/src
# ./build.sh tools
# ./build.sh distribution
# cd /usr/pkg/share/basepkg
# ./basepkg.sh pkg
```

- How to build kernel packages

```
# cd /usr/src
# ./build.sh kernel=GENERIC
# cd /usr/pkg/share/basepkg
# ./basepkg.sh kern
```

basepkg (9/12)

- It’s easy to add or delete the specific base package by using **pkg_*** tools since the package format is same as **pkgsrc**(7) one.
  - **pkg_add**(1) – Install the package
  - **pkg_delete**(1) – Remove the package

- To avoid confliction between pkgsrc and basepkg packages, we should specify the other database path such as

```
# pkg_add -K /var/db/basepkg base-sys-root
```

basepkg (10/12)

- Currently in using raw **pkg_*** tools to manipulate packages, we need to be very careful to handle **etc** package.
  - E.g. **etc-sys-etc-7.1.tgz**
  - Because it overwrites files under the **/etc** directory.

- To avoid this disaster, once we extract the contents in another directory and running install script.

```
# pkg_add -K /var/db/basepkg -P /tmp/basepkg etc-sys-etc.7.1.tgz
```

- We should prepare a wrapper for users not to handle raw **pkg_*** tools.

basepkg (11/12)

- We have compared the installation time between tarball extraction and using package made by **basepkg**.
  1. Fetch a tarball "**`games.tgz`**" from `ftp.jp.netbsd.org`, then extract it.
  2. Install all packages beginning with "**`games`**" to system from `basepkg.netbsd.fml.org`
  3. Install one "**`games-games-bin`**" package to system from `basepkg.netbsd.fml.org`

- Where `basepkg.netbsd.fml.org` is an experimental base package distribution server we build and operate.

basepkg (12/12)

Test	Real Time (s)	User Time (s)	System Time (s)
tarball	7.2374	0.2267	0.8433
All packages	19.2955	0.9457	1.1725
One package	3.4656	0.0838	0.0924

- Only when we update a few packages in the system, the process is comparable to the tarball extracting.

- In almost cases under normal operation, we replace only a few small parts for rapid security update.

Demonstration

- wip/basepkg
  - make install clean clean-depends
- cd /usr/pkg/basepkg
- Run basepkg.sh.
- pkg_add/pkg_delete some package.

Demonstration

- pkgtools/pkgin
- Create repository.
- Install package using pkgin.

Contents

Discussion (1/8)

- We summarize changes and improvements from last year.
    - Import to **pkgsrc-wip** repository.
        - **wip/basepkg**
    - Update database (under **sets/**).
        - Fix PR#46937 by Lloyd Parkes at 2012.
    - Enhance meta data
        - +DEINSTALL, +INSTALL, +PRESERVE, +SIZE_PKG, +SIZE_ALL
    - Cross build support
    - Multi platform support
        - We have verified **basepkg.sh** can run on Ubuntu 17.04 amd64.

Discussion (2/8)

- There are a lot of technical issues to resolve as follows.

1. basepkg processing speed.
    - **basepkg.sh** slower than **build.sh syspkgs**.
    - We must need to try better shell coding technique.
    - We should not use **for** or **while** loop as could as possible, instead use internal loops such as **find**(1) and **grep**(1).

2. basepkg database (under **sets/**) maintenance.
    - It looks **descrs** and **comments** has been incomplete.

Discussion (3/8)

- There are a lot of technical issues to resolve as follows.

3. A wrapper convenient for users.
    - Set database location.
    - Provide alias mapping for ambiguous package names.
        - **`# wrapper install openssh`** --> **`# wrapper install base-secsh-bin`**

4. Integrated system management support.
    - **`pkg-vulnerabilities`**
    - **`pkgsrc/pkgtools/pkgin`**

Discussion (4/8)

- How to use **pkgin**(1) for base package.
  1. Install **pkgtools/pkgin**
  2. Edit **/usr/pkg/etc/pkgin/repositories.conf**
     ```
     # echo “file:///path/to/basepkg/packages/7.1.1/amd64-x86_64/” ¥
     >> /usr/pkg/etc/pkgin/repositories.conf
     ```
  3. `# cd /path/to/basepkg/packages/7.1.1/amd64-x86_64`
  4. `# pkg_info --X *.tgz > pkg_summary`
  5. `# gzip pkg_summary`
  6. `# pkgin in base-sys-root`

Discussion (5/8)

- We estimated of base package distribution.

- In the case of building source and package by distribution server …….

- Current VPS Case
  - basepkg.netbsd.fml.org which runs on SAKURA VPS 2GB 3Core 200GB Disk.
  - We providing 30 architectures in NetBSD 7 stable.
    - One architecture requires 5GB Disk space. So, the upper limit of 30 architectures are restricted by this storage limit to run build.sh.
  - Building process costs about 1 hour per target.
  - If we can run processes parallelly per CPU core, we need 10 hour to prepare 30 architectures.

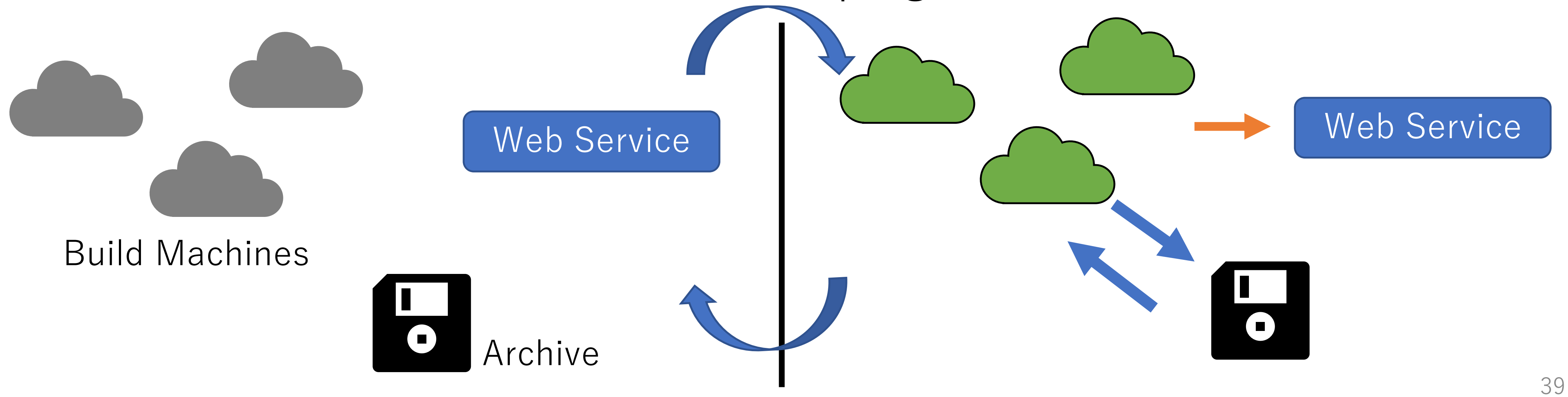
Discussion (6/8)

- Cloud Case (The evaluation is underway)
  - Cloud service is more suitable for intermittent work like this.
  - The updates for stable branches are rare,
    so we don’t need to build package daily.

- If we run this building process only when
  a NetBSD security advisory is released and
  the target can be restricted to stable branches,
  modern cloud service is more proper than
  the current VPS service.

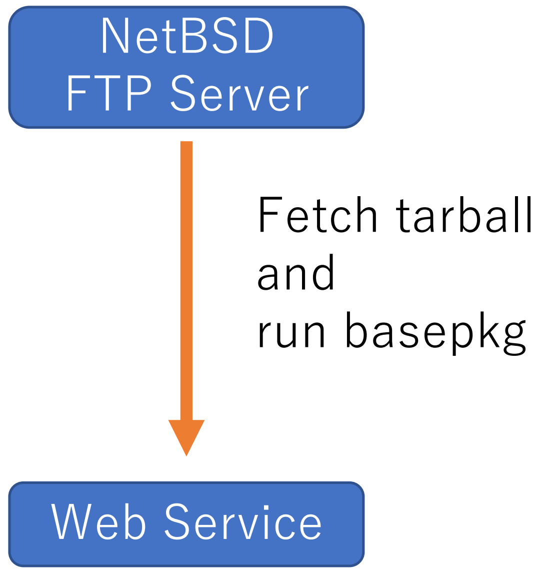
Discussion (7/8)

- In the case of cloud service, we assume the following usage:
  - Normally the build process does not run. The low cost cloud archive holds the built data.
  - On demand, we wake up the cloud service, extract the built data from the archive, build packages, update web service, re-archive the built data and make the cloud sleep again.



Discussion (8/8)

- In the case of building only package by distribution server ….
- Today it looks NetBSD daily build system can prepare daily binaries for some branches.
- Hence basepkg distribution server can fetch the tarballs and build base packages based on them.
- We hope to operate package distribution server at a low cost but only for latest branches.



Contents

Conclusion (1/1)

- We have developed third-party framework “**basepkg**” to packaging NetBSD base system.
- It’s shown that this framework provides more granular and faster update of NetBSD base system and useful functions for users.
- However, we have a lot of issues to resolve for realistic system operations, so we need to continue dogfooding and development.