NetBSD Base System Packaging Using pkg_* Tools

Yuuki Enomoto and Ken'ichi Fukamachi∗
Affiliations and address
Chitose Institute of Science and Technology, e-mail: m2160020@photon.chitose.ac.jp
∗Affiliations and address
Chitose Institute of Science and Technology, e-mail: k-fukama@photon.chitose.ac.jp

Abstract

Unix operating system (OS) such as Linux distributions and BSD is widely used mainly as servers and embedded systems. Today, on Unix OS, speedy update of the system is required especially to make the system up-to-date for security. For rapid update, OS should be fine granular. So, we need to package the OS base system where "package" implies a set of software, documentation, configuration files and package's meta data and "packaging" implies that an OS consists of a lot of small packages to add or delete on demand. In Linux distributions such as Debian, Ubuntu and Red Hat Enterprise Linux, the userland programs are already divided into many small packages. On the other hand, BSD Unix such as FreeBSD and NetBSD are behind the curve on the base system packaging. We focus on NetBSD for portability and license. We have developed a software called "basepkg" for NetBSD base system packaging. It makes the update process of NetBSD base system more granular and faster.

Key words: Unix, NetBSD, Open Source Software, System Management

1 Introduction

1.1 Unix operating system's status on business

Unix operating system (OS) such as Linux distributions and BSD-Unix is widely used mainly as commercial servers, embedded systems and software products all over the world. According to data compiled by The Linux Foundation, Linux application deployments have risen during from 2011 to 2014, from 65 percent to 79 percent; and Linux has a 75 percent share in primary cloud platform in 2014 [1]. Also, BSD-Unix is used for embedded systems like RICOH printer [2] and any software products.

1.2 Use of packages for OS management

It is considered to be preferable that an OS consists of a lot of parts to be added or removed on demand since we can update the system easily and frequently to fix security holes speedy. For that reason, many Unix OS use "package" systems to manage each system where this "package" implies a container which consists of software, documentation, configuration files and this package's meta data required to operate in installation and de-installation. Each package role, format and manager differs from one Unix OS to another. Table 1 shows a list of OS, package's format and package manager.

Table 1: List of OS, package's format and package manager

Name	package's format	management software
FreeBSD	txz	pkg
NetBSD	tgz	pkg_install
Debian	deb	apt
Red Hat	rpm	yum
openSUSE	rpm	zypper

Major Linux distributions such as Debian and Red Hat Enterprise Linux are already divided into many small packages. These OS's can manage its own base system and third-party software through

its package manager where the base system means the minimum set officially provided and maintained by each OS project.

On the other hand, BSD Unix such as FreeBSD and NetBSD have each package framework e.g. `ports`(7) and `pkgsrc`(7), but they have been used only for third-party software management.

1.3 Open Source Software Licenses

Open Source Software (OSS) is software distributed with the source code that anyone can copy, modify, and (re-)distribute. Many software that constitute Linux distributions and BSD Unix are OSS. OSS is protected by a specific license. This section describes the difference of licenses between Linux distribution and BSD Unix.

"GNU General Public License" (GPL) was created by the Free Software Foundation (FSF). The typical examples of software under the GPL are Linux kernel, Emacs, and GCC. The GPL demands the following restrictions:

1. Permission to run the software.
2. Permission to copy of the source code and software,
3. Permission to modify the source code.
4. Permission to distribute the software.
5. Software that used GPL's source code applies same license.

The source code using GPL is supposed to be public irrespective of commercial or non-commercial use. For that reason, GPL violation may be prosecuted. In a famous case, Software Freedom Law Center (SFLC) conducted a lawsuit to 14 electronics manufacturers including Best Buy, JVC, and Samsung-infringed concerning GPL violations of the BusyBox software[3]. A paper published by Wasabi Systems, Inc.[4] says

> In sum, the effect of the GPL depends on its application. [...] To desktop and server users of Linux, it is largely irrelevant. For commercial embedded companies, as for any other firms in which kernel modification and driver modifications are important, the GPL can be catastrophic.

"BSD License" was created by the Regents of the University of California. The typical example of software under the BSD License is BSD Unix. The BSD License (2-clause) demands only two restrictions:

1. Retain the copyright notice and provision of BSD License.
2. Stipulate be as-is software.

Today, the most famous Unix OS is Linux distributions. However for commercial use of Unix, it is preferable to consider the adoption of BSD Unix since GPL may be potential lawsuit.

1.4 Status of BSD base system packaging

The base system packaging on BSD Unix is better to implement the rapid granular update and to avoid the potential lawsuit described above. In this section, we summarize the current status of base system packaging on a few BSD Unix.

FreeBSD was born at December of 1993. FreeBSD is known to support huge number of applications; ZFS and BeHyve are especially well known. PkgBase (packaged base) became a beta feature in the FreeBSD 11 branch with revision number 298107 at April 16, 2016 by gjb@. From the point of PkgBase view, FreeBSD base system comprises around 795 packages in the case of amd64 architecture by default. To manage the packages, FreeBSD uses a new package manager called "pkg" introduced in FreeBSD version 11.0-RELEASE.

NetBSD has another framework called "syspkgs". It was imported to the NetBSD base system at January 8, 2002 by jwise@. However judging from log messages of distrib/syspkg directory in the source tree, syspkgs has not been maintained since February 21, 2010.

2 Our Goals

NetBSD is well known to be portable so that NetBSD needs to cover from too slow to modern fast architectures. On slow machines, it is hard to rebuild the system frequently for security update. To relieve the burden, the base system packaging is useful. It introduces fine granularity of the base system. Hence it enables the rapid update of a part of the base system on resource limited machines.

Such a system called "syspkg" is incompletely implemented on NetBSD. It is also stagnant these years. There are several technical problems. For example, the package made by syspkgs is primitive compared to one by pkgsrc. It is hard to directly fix syspkgs framework which consists of a lot of makefiles, scripts and undocumented data. For this reason, we have developed another base packaging mechanism as a third party software.

3 Basepkg

3.1 What is Basepkg

We have developed a new framework "basepkg" that can package NetBSD base system instead of syspkgs. The basepkg is an open source software distributed under the BSD License, and published on `https://github.com/user340/basepkg`. The behavior of basepkg is similar to syspkgs. The comparison between them are shown at table 2.

Table 2: comparison between syspkgs and basepkg

feature	syspkgs	basepkg
language	Makefile and Bourne shell	Bourne shell
install script	none	available
kernel package	none	supporting GENERIC kernel
imported to	official source tree	pkgsrc-wip

From the aspect of portability, basepkg is written in Bourne shell, which is same as syspkgs.

In the table 2, the term "install script" implies a hook which can be executed within package installation or de-installation process.

The formal degree is different. Our basepkg was developed as a third-party software, but syspkgs is a part of the base system. The syspkgs building process can be hooked within "build.sh" which is the top level program of official NetBSD source building system.

3.2 How to use Basepkg

The basepkg requires NetBSD source tree and pkg_* tools.

Firstly, build the NetBSD.

```
# cd /usr/src
# ./build.sh -O ../obj -T ../tools tools
# ./build.sh -O ../obj -T ../tools distribution
# ./build.sh -O ../obj -T ../tools kernel=GENERIC
```

Secondly, run the `basepkg.sh` with "pkg" and "kern" options. Baspkg generates packages at `packages/[NetBSD version]/[MACHINE]-[MACHINE_ARCH]` directory. For example, if you run `basepkg.sh` on `NetBSD-7.1/amd64`, packages are generated at `packages/7.1/amd64-x86_64` directory by default.

```
# cd /path/to/basepkg
# ./basepkg.sh pkg
# ./basepkg.sh kern
```

The generated packages can be installed to the base system using `pkg_add`(1). In contrast, installed packages can be removed from the base system using `pkg_delete`(1).

```
# cd ./packages/7.1/amd64-x86_64
# pkg_add    -K /var/db/basepkg games-games-bin
# pkg_delete -K /var/db/basepkg games-games-bin
```

4 Comparison between Old and New Update Processes

Packaging implies that an OS consists of a set of small files, so that the OS update process to add or delete small files must be faster. However the package size to add or delete is not proportional to the speed of update process since packaging introduces several overheads e.g. resolution of calculation dependencies, execution of install scripts, and other processes.

We compared the installation time between the traditional (tarball) and our new methods (basepkg). We processed the following updates 100 times on NetBSD-7.1/amd64. We used *time*(1) command to measure the process time. The target category we used is "game" since "game" category is not mission critical.

1. Fetch a "games.tgz" from `ftp.jp.netbsd.org/pub/NetBSD/NetBSD-7.1/amd64/binary/sets/`, then extract to `./tmp directory`.
2. Install all packages beginning with "games-" to system from `basepkg.netbsd.fml.org/pub/NetBSD/basepkg/7.1/amd64-x86_64`
3. Install one "games-games-bin" package to system from `basepkg.netbsd.fml.org/pub/NetBSD/basepkg/7.1/amd64-x86_64`

Table 3 shows the average time of the result.

Table 3: Average of process time in installation

Test	real time (s)	user time (s)	system time (s)
1	7.2374	0.2267	0.8443
2	19.2955	0.9457	1.1725
3	3.4656	0.0838	0.0924

Table 3 verifies the installation using basepkg is faster than traditional one. However it is not faster than we expected because of overheads mentioned above. Only when we update a few packages in the system, the process is faster than the traditional one (extraction of a large tar ball).

5 Conclusion

BSD Unix is suited to commercial products and service more than Linux distributions. We have developed another framework "basepkg" to package NetBSD base system. Using this framework, we can manage the base system more gradually than in the traditional way. However we have a lot to do for real system operation, so we need to continue dogfooding and development.

References

[1] Linux.com "2014 Enterprise End User Report" Retrieved October 25, 2017 from `www.linux.com/publications/2014-enterprise-end-user-report`

[2] The NetBSD Project "Products based on NetBSD" Retrieved October 25, 2017 from `www.netbsd.org/gallery/products.html\#richo`

[3] Software Freedom Law Center "Best Buy, Samsung, Westinghouse, And Eleven Other Brands Named In SFLC Lawsuit" Retrieved October 28, 2017 from `www.softwarefreedom.org/news/2009/dec/14/busybox-gpl-lawsuit/`

[4] Wasabi Systems "BSD or Linux: Which Unix is best for embedded applications?" Retrieved October 27, 2017 from `www.wasabisystems.com/wp-content/uploads/2016/06/Linux_or_BSD-1.pdf`