IBLIOqqq

# FreeBSD Release Engineering


## Murray Stokely

**murray@FreeBSD.org**



**$FreeBSD: doc/en_US.ISO8859-1/articles/releng/article.sgml,v 1.83 2009/06/24 22:25:26 keramida Exp $**



**FreeBSD is a registered trademark of the FreeBSD Foundation.**
**CVSup is a registered trademark of John D. Polstra.**
**Intel, Celeron, EtherExpress, i386, i486, Itanium, Pentium, and Xeon are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.**
**XFree86 is a trademark of The XFree86 Project, Inc.**
**Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this document, and the FreeBSD Project was aware of the trademark claim, the designations have been followed by the “™” or the “®” symbol.**



This paper describes the approach used by the FreeBSD release engineering team to make production quality releases of the FreeBSD Operating System. It details the methodology used for the official FreeBSD releases and describes the tools available for those interested in producing customized FreeBSD releases for corporate rollouts or commercial productization.

# Table of Contents



## 1 Introduction

The development of FreeBSD is a very open process. FreeBSD is comprised of contributions from thousands of people around the world. The FreeBSD Project provides anonymous CVS[1] access to the general public so that others can have access to log messages, diffs (patches) between development branches, and other productivity enhancements that formal source code management provides. This has been a huge help in attracting more talented developers to FreeBSD. However, I think everyone would agree that chaos would soon manifest if write access was opened up to everyone on the Internet. Therefore only a "select" group of nearly 300 people are given write access to the CVS repository. These *committers[5]* are responsible for the bulk of FreeBSD development. An elected *core-team[6]* of very senior developers provides some level of direction over the project.

The rapid pace of FreeBSD development leaves little time for polishing the development system into a production quality release. To solve this dilemma, development continues on two parallel tracks. The main development branch is the *HEAD* or *trunk* of our CVS tree, known as "FreeBSD-CURRENT" or "-CURRENT" for short.

A more stable branch is maintained, known as "FreeBSD-STABLE" or "-STABLE" for short. Both branches live in a master CVS repository in California and are replicated via **CVSup**[2] to mirrors all over the world. FreeBSD-CURRENT[7] is the "bleeding-edge" of FreeBSD development where all new changes first enter the system. FreeBSD-STABLE is the development branch from which major releases are made. Changes go into this branch at a different pace, and with the general assumption that they have first gone into FreeBSD-CURRENT and have been thoroughly tested by our user community.

In the interim period between releases, monthly snapshots are built automatically by the FreeBSD Project build machines and made available for download from ftp://ftp.freebsd.org/pub/FreeBSD/snapshots/. The widespread availability of binary release snapshots, and the tendency of our user community to keep up with -STABLE development with CVSup and "`make world`"[7] helps to keep FreeBSD-STABLE in a very reliable condition even before the quality assurance activities ramp up pending a major release.

Bug reports and feature requests are continuously submitted by users throughout the release cycle. Problems reports are entered into our **GNATS**[8] database through email, the send-pr(1) application, or via the web interface provided at http://www.FreeBSD.org/send-pr.html. In addition to the multitude of different technical mailing lists about FreeBSD, the FreeBSD Quality Assurance mailing list (http://lists.FreeBSD.org/mailman/listinfo/freebsd-qa) provides a forum for discussing the finer points of "release-polishing".

To service our most conservative users, individual release branches were introduced with FreeBSD 4.3. These release branches are created shortly before a final release is made. After the release goes out, only the most critical security

fixes and additions are merged onto the release branch. In addition to source updates via CVS, binary patchkits are available to keep systems on the *RELENG_X_Y* branches updated.

### 1.1 What this article describes

The following sections of this article describe:

Section 2

The different phases of the release engineering process leading up to the actual system build.

Section 3

The actual build process.

Section 5

How the base release may be extended by third parties.

Section 6

Some of the lessons learned through the release of FreeBSD 4.4.

Section 7

Future directions of development.

## 2 Release Process

New releases of FreeBSD are released from the -STABLE branch at approximately four month intervals. The FreeBSD release process begins to ramp up 45 days before the anticipated release date when the release engineer sends an email to the development mailing lists to remind developers that they only have 15 days to integrate new changes before the code freeze. During this time, many developers perform what have become known as “MFC sweeps”. MFC stands for “Merge From CURRENT” and it describes the process of merging a tested change from our -CURRENT development branch to our -STABLE branch.

### 2.1 Code Review

Thirty days before the anticipated release, the source repository enters a “code slush”. During this time, all commits to the -STABLE branch must be approved by the Release Engineering Team <re@FreeBSD.org>. The kinds of changes that are allowed during this 15 day period include:

- Bug fixes.
- Documentation updates.
- Security-related fixes of any kind.
- Minor changes to device drivers, such as adding new Device IDs.
- Any additional change that the release engineering team feels is justified, given the potential risk.

After the first 15 days of the code slush, a *release candidate* is released for widespread testing and the code enters a "code freeze" where it becomes much harder to justify new changes to the system unless a serious bug-fix or security issue is involved. During the code freeze, at least one release candidate is released per week, until the final release is ready. During the days leading to the final release, the release engineering team is in constant communication with the security-officer team, the documentation maintainers, and the port maintainers, to ensure that all of the different components required for a successful release are available.

## 2.2 Final Release Checklist

When several release candidates have been made available for widespread testing and all major issues have been resolved, the final release "polishing" can begin.

### 2.2.1 Creating the Release Branch

As described in the introduction, the `RELENG_X_Y` release branch is a relatively new addition to our release engineering methodology. The first step in creating this branch is to ensure that you are working with the newest version of the `RELENG_X` sources that you want to branch *from*.

```
/usr/src# cvs update -rRELENG_4 -P -d
```

The next step is to create a branch point *tag*, so that diffs against the start of the branch are easier with CVS:

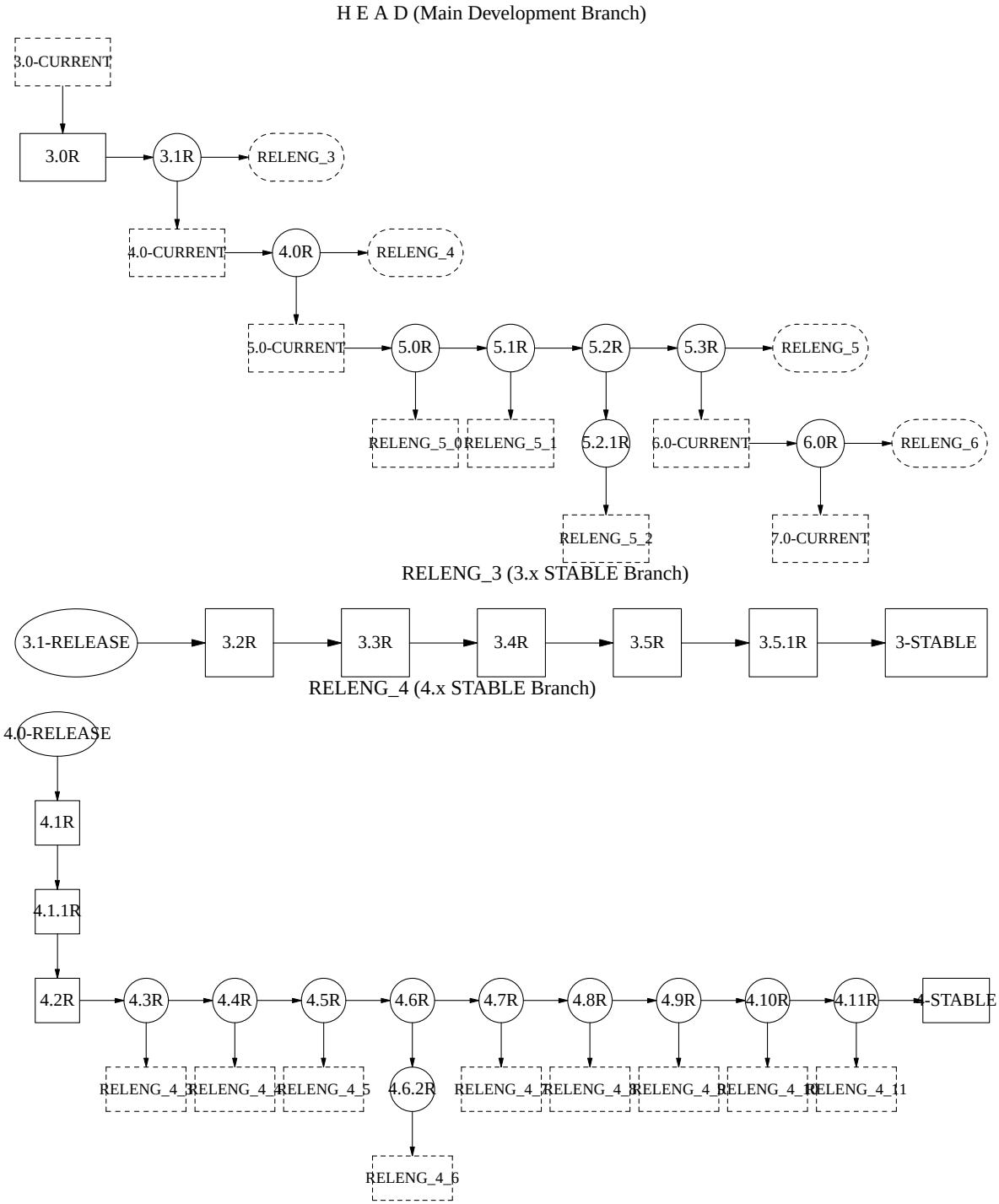
```
/usr/src# cvs rtag -rRELENG_4 RELENG_4_8_BP src
```

And then a new branch tag is created with:

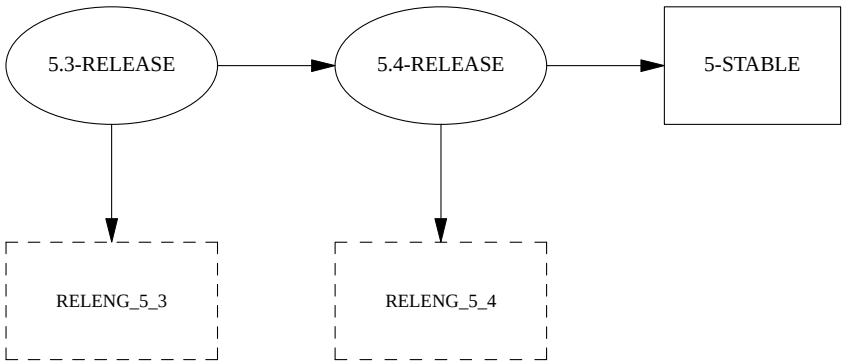
```
/usr/src# cvs rtag -b -rRELENG_4_8_BP RELENG_4_8 src
```

**Note:** *The `RELENG_*` tags are restricted for use by the CVS-meisters and release engineers.*

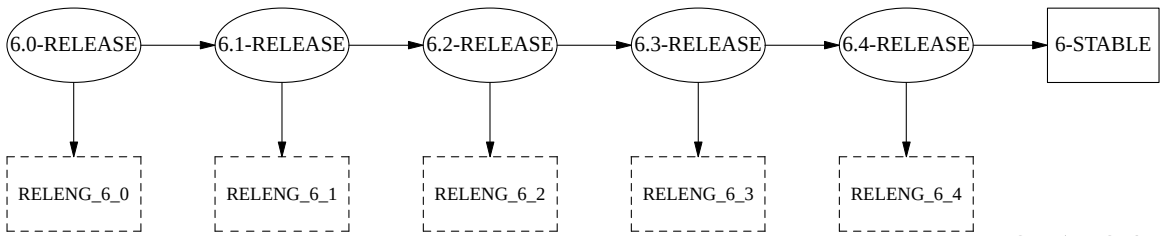
A "*tag*" is CVS vernacular for a label that identifies the source at a specific point in time. By tagging the tree, we ensure that future release builders will always be able to use the same source we used to create the official FreeBSD Project releases.


H E A D (Main Development Branch)
3.0-CURRENT
3.0R
3.1R
RELENG_3
4.0-CURRENT
4.0R
RELENG_4
5.0-CURRENT
5.0R
5.1R
5.2R
5.3R
RELENG_5
RELENG_5_0
RELENG_5_1
5.2.1R
6.0-CURRENT
6.0R
RELENG_6
RELENG_5_2
7.0-CURRENT
RELENG_3 (3.x STABLE Branch)
3.1-RELEASE
3.2R
3.3R
3.4R
3.5R
3.5.1R
3-STABLE
RELENG_4 (4.x STABLE Branch)
4.0-RELEASE
4.1R
4.1.1R
4.2R
4.3R
4.4R
4.5R
4.6R
4.7R
4.8R
4.9R
4.10R
4.11R
4-STABLE
RELENG_4_3
RELENG_4_4
RELENG_4_5
4.6.2R
RELENG_4_7
RELENG_4_8
RELENG_4_9
RELENG_4_10
RELENG_4_11
RELENG_4_6

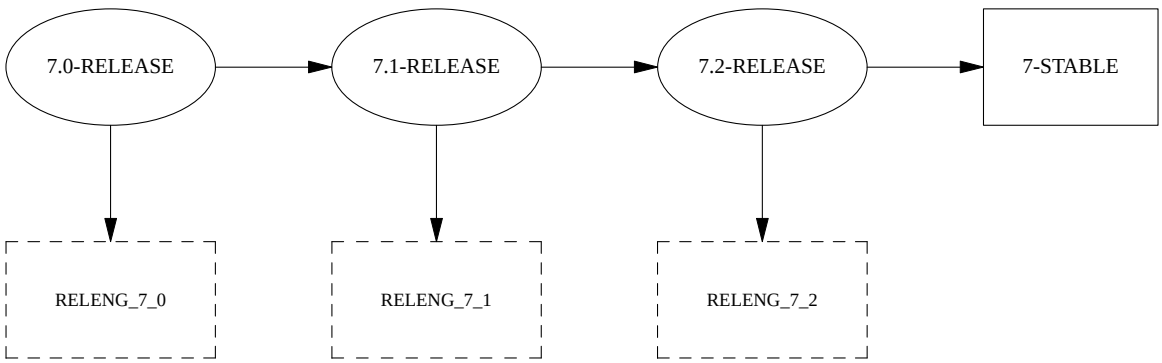
RELENG_5 (5.x STABLE Branch)




RELENG_6 (6.x STABLE Branch)




RELENG_7 (7.x STABLE Branch)




### 2.2.2 Bumping up the Version Number

Before the final release can be tagged, built, and released, the following files need to be modified to reflect the correct version of FreeBSD:

- `doc/en_US.ISO8859-1/books/handbook/mirrors/chapter.sgml`
- `doc/en_US.ISO8859-1/books/porters-handbook/book.sgml`
- `doc/share/sgml/freebsd.ent`
- `src/Makefile.inc1`
- `src/UPDATING`
- `src/gnu/usr.bin/groff/tmac/mdoc.local`
- `src/release/Makefile`

- `src/release/doc/en_US.ISO8859-1/share/sgml/release.dsl`
- `src/release/doc/share/examples/Makefile.relnotesng`
- `src/release/doc/share/sgml/release.ent`
- `src/share/examples/cvsup/standard-supfile`
- `src/sys/conf/newvers.sh`
- `src/sys/sys/param.h`
- `src/usr.sbin/pkg_install/add/main.c`
- `www/en/docs/man.sgml`
- `www/en/cgi/ports.cgi`
- `ports/Tools/scripts/release/config`

The release notes and errata files also need to be adjusted for the new release (on the release branch) and truncated appropriately (on the stable/current branch):

- `src/release/doc/en_US.ISO8859-1/relnotes/common/new.sgml`
- `src/release/doc/en_US.ISO8859-1/errata/article.sgml`

**Sysinstall** should be updated to note the number of available ports and the amount of disk space required for the Ports Collection[4]. This information is currently kept in `src/usr.sbin/sysinstall/dist.c`.

After the release has been built, a number of file should be updated to announce the release to the world.

- `doc/share/images/articles/releng/branches-relengX.pic`
- `www/share/sgml/advisories.xml`
- `www/share/sgml/includes.release.sgml`
- `www/share/sgml/includes.release.xsl`
- `www/en/releases/*`
- `www/en/releng/index.sgml`
- `www/en/news/news.xml`
- `www/en/search/web.atoz`
- `src/share/misc/bsd-family-tree`

### 2.2.3 Preparing a new major release branch (RELENG_*x*)

When a new major release branch, such as `RELENG_6` is branched from HEAD, some additional files must be updated before releases can be made from this new branch.

- `src/share/examples/cvsup/stable-supfile` - must be updated to point to the new -STABLE branch, when applicable.

### 2.2.4 Creating Release Tags

When the final release is ready, the following command will create the `RELENG_4_8_0_RELEASE` tag.

```
/usr/src# cvs rtag -rRELENG_4_8 RELENG_4_8_0_RELEASE src
```

The Documentation and Ports managers are responsible for tagging the respective trees with the `RELEASE_4_8_0` tag.

Occasionally, a last minute fix may be required *after* the final tags have been created. In practice this is not a problem, since CVS allows tags to be manipulated with `cvs tag -d` *`tagname filename`*. It is very important that any last minute changes be tagged appropriately as part of the release. FreeBSD releases must always be reproducible. Local hacks in the release engineer's environment are not acceptable.

# 3 Release Building

FreeBSD "releases" can be built by anyone with a fast machine and access to a source repository. (That should be everyone, since we offer anonymous CVS! See The Handbook for details.) The *only* special requirement is that the vn(4) device must be available. (On -CURRENT, this device has been replaced by the new md(4) memory disk driver.) If the device is not loaded into your kernel, then the kernel module should be automatically loaded when vnconfig(8) is executed during the boot media creation phase. All of the tools necessary to build a release are available from the CVS repository in `src/release`. These tools aim to provide a consistent way to build FreeBSD releases. A complete release can actually be built with only a single command, including the creation of ISO images suitable for burning to CDROM, installation floppies, and an FTP install directory. This command is aptly named `make release`.

## 3.1 `make release`

To successfully build a release, you must first populate `/usr/obj` by running `make world` or simply `make buildworld`. The release target requires several variables be set properly to build a release:

- `CHROOTDIR` - The directory to be used as the chroot environment for the entire release build.
- `BUILDNAME` - The name of the release to be built.
- `CVSROOT` - The location of a CVS Repository.
- `RELEASETAG` - The CVS tag corresponding to the release you would like to build.

If you do not already have access to a local CVS repository, then you may mirror one with CVSup (http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/synching.html#CVSUP). The supplied supfile, `/usr/share/examples/cvsup/cvs-supfile`, is a useful starting point for mirroring the CVS repository.

If `RELEASETAG` is omitted, then the release will be built from the `HEAD` (a.k.a. -CURRENT) branch. Releases built from this branch are normally referred to as "-CURRENT snapshots".

There are many other variables available to customize the release build. Most of these variables are documented at the top of `src/release/Makefile`. The exact command used to build the official FreeBSD 4.7 (x86) release was:

```
make release CHROOTDIR=/local3/release \
       BUILDNAME=4.7-RELEASE \
```

```
CVSROOT=/host/cvs/usr/home/ncvs \
RELEASETAG=RELENG_4_7_0_RELEASE
```

The release `Makefile` can be broken down into several distinct steps.

- Creation of a sanitized system environment in a separate directory hierarchy with "`make installworld`".
- Checkout from CVS of a clean version of the system source, documentation, and ports into the release build hierarchy.
- Population of `/etc` and `/dev` in the chrooted environment.
- chroot into the release build hierarchy, to make it harder for the outside environment to taint this build.
- `make world` in the chrooted environment.
- Build of Kerberos-related binaries.
- Build `GENERIC` kernel.
- Creation of a staging directory tree where the binary distributions will be built and packaged.
- Build and installation of the documentation toolchain needed to convert the documentation source (SGML) into HTML and text documents that will accompany the release.
- Build and installation of the actual documentation (user manuals, tutorials, release notes, hardware compatibility lists, and so on.)
- Build of the "crunched" binaries used for installation floppies.
- Package up distribution tarballs of the binaries and sources.
- Create the boot media and a "fixit" floppy.
- Create FTP installation hierarchy.
- *(optionally)* Create ISO images for CDROM/DVD media.

For more information about the release build infrastructure, please see release(7).

## 3.2 Building XFree86™

**XFree86™** is an important component for many desktop users. Prior to FreeBSD 4.6-RELEASE, releases used XFree86 3.*X* by default. The easiest way to build these versions is to use the `src/release/scripts/X11/build_x.sh` script. This script requires that XFree86 and Tcl/Tk already be installed on the build host. After compiling the necessary X servers, the script will package all of the files into tarballs that sysinstall(8) expects to find in the `XF86336` directory of the installation media.

Beginning with FreeBSD 4.6-RELEASE, sysinstall(8) installs XFree86 4.*X* by default, as a set of "normal" packages. These can either be the packages generated by the package-building cluster or packages built from an appropriately tagged ports tree.

Beginning with FreeBSD 5.3-RELEASE, sysinstall(8) installs Xorg packages instead of XFree86 packages by default.

**Note:** It is important to remove any site-specific settings from `/etc/make.conf`. For example, it would be unwise to distribute binaries that were built on a system with `CPUTYPE` set to a specific processor.

## 3.3 Contributed Software ("ports")

The FreeBSD Ports collection (http://www.FreeBSD.org/ports) is a collection of over 20,000 third-party software packages available for FreeBSD. The Ports Management Team <`portmgr@FreeBSD.org`> is responsible for maintaining a consistent ports tree that can be used to create the binary packages that accompany official FreeBSD releases.

The release engineering activities for our collection of third-party packages is beyond the scope of this document. A separate article, The Release Engineering of Third Party Packages (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/releng-packages/article.html), covers this topic in depth.

## 3.4 Release ISOs

Starting with FreeBSD 4.4, the FreeBSD Project decided to release all four ISO images that were previously sold on the *BSDi/Wind River Systems/FreeBSD Mall* "official" CDROM distributions. Each of the four discs must contain a `README.TXT` file that explains the contents of the disc, a `CDROM.INF` file that provides meta-data for the disc so that sysinstall(8) can validate and use the contents, and a `filename.txt` file that provides a manifest for the disc. This *manifest* can be created with a simple command:

```
/stage/cdrom# find . -type f | sed -e 's/^\.\///' | sort > filename.txt
```

The specific requirements of each CD are outlined below.

### 3.4.1 Disc 1

The first disc is almost completely created by `make release`. The only changes that should be made to the `disc1` directory are the addition of a `tools` directory, **XFree86**, and as many popular third party software packages as will fit on the disc. The `tools` directory contains software that allow users to create installation floppies from other operating systems. This disc should be made bootable so that users of modern PCs do not need to create installation floppy disks.

If an alternate version of XFree86 is to be provided, then sysinstall(8) must be updated to reflect the new location and installation instructions. The relevant code is contained in `src/usr.sbin/sysinstall`. Specifically, the files `dist.c`, `menus.c`, and `config.c` will need to be updated.

### 3.4.2 Disc 2

The second disc is also largely created by `make release`. This disc contains a "live filesystem" that can be used from sysinstall(8) to troubleshoot a FreeBSD installation. This disc should be bootable and should also contain a compressed copy of the CVS repository in the `CVSROOT` directory and commercial software demos in the `commerce` directory.

### 3.4.3 Discs 3 and 4

The remaining two discs contain additional software packages for FreeBSD. The packages should be clustered so that a package and all of its *dependencies* are included on the same disc. More information about the creation of these discs is provided in the The Release Engineering of Third Party Packages (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/releng-packages/article.html) article.

### 3.4.4 Multi-volume support

**Sysinstall** supports multiple volume package installations. This requires that each disc have an `INDEX` file containing all of the packages on all volumes of a set, along with an extra field that indicates which volume that particular package is on. Each volume in the set must also have the `CD_VOLUME` variable set in the `cdrom.inf` file so that sysinstall can tell which volume is which. When a user attempts to install a package that is not on the current disc, sysinstall will prompt the user to insert the appropriate one.

# 4 Distribution

## 4.1 FTP Sites

When the release has been thoroughly tested and packaged for distribution, the master FTP site must be updated. The official FreeBSD public FTP sites are all mirrors of a master server that is open only to other FTP sites. This site is known as `ftp-master`. When the release is ready, the following files must be modified on `ftp-master`:

`/pub/FreeBSD/releases/`*`arch`*`/`*`X.Y`*`-RELEASE/`

The installable FTP directory as output from `make release`.

`/pub/FreeBSD/ports/`*`arch`*`/packages-`*`X.Y`*`-release/`

The complete package build for this release.

`/pub/FreeBSD/releases/`*`arch`*`/`*`X.Y`*`-RELEASE/tools`

A symlink to `../../../tools`.

`/pub/FreeBSD/releases/`*`arch`*`/`*`X.Y`*`-RELEASE/packages`

A symlink to `../../../ports/`*`arch`*`/packages-`*`X.Y`*`-release`.

`/pub/FreeBSD/releases/`*`arch`*`/ISO-IMAGES/`*`X.Y`*`/`*`X.Y`*`-RELEASE-`*`arch`*`-*.iso`

The ISO images. The "*" is `disc1`, `disc2`, etc. Only if there is a `disc1` and there is an alternative first installation CD (for example a stripped-down install with no windowing system) there may be a `mini` as well.

For more information about the distribution mirror architecture of the FreeBSD FTP sites, please see the Mirroring FreeBSD (http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/hubs/) article.

It may take many hours to two days after updating `ftp-master` before a majority of the Tier-1 FTP sites have the new software depending on whether or not a package set got loaded at the same time. It is imperative that the release engineers coordinate with the FreeBSD mirror site administrators

(http://lists.FreeBSD.org/mailman/listinfo/mirror-announce) before announcing the general availability of new software on the FTP sites. Ideally the release package set should be loaded at least four days prior to release day. The release bits should be loaded between 24 and 48 hours before the planned release time with "other" file permissions turned off. This will allow the mirror sites to download it but the general public will not be able to download it from the mirror sites. Mail should be sent to FreeBSD mirror site administrators (http://lists.FreeBSD.org/mailman/listinfo/mirror-announce) at the time the release bits get posted saying the release has been staged and giving the time that the mirror sites should begin allowing access. Be sure to include a time zone with the time, for example make it relative to GMT.

### 4.2 CD-ROM Replication

Coming soon: Tips for sending FreeBSD ISOs to a replicator and quality assurance measures to be taken.

## 5 Extensibility

Although FreeBSD forms a complete operating system, there is nothing that forces you to use the system exactly as we have packaged it up for distribution. We have tried to design the system to be as extensible as possible so that it can serve as a platform that other commercial products can be built on top of. The only "rule" we have about this is that if you are going to distribute FreeBSD with non-trivial changes, we encourage you to document your enhancements! The FreeBSD community can only help support users of the software we provide. We certainly encourage innovation in the form of advanced installation and administration tools, for example, but we cannot be expected to answer questions about it.

### 5.1 Creating Customized Boot floppies

Many sites have complex requirements that may require additional kernel modules or userland tools be added to the installation floppies. The "quick and dirty" way to accomplish this would be to modify the staging directory of an existing `make release` build hierarchy:

- Apply patches or add additional files inside the chroot release build directory.
- `rm ${CHROOTDIR}/usr/obj/usr/src/release/release.[59]`
- rebuild sysinstall(8), the kernel, or whatever parts of the system your change affected.
- `chroot ${CHROOTDIR} ./mk floppies`

New release floppies will be located in `${CHROOTDIR}/R/stage/floppies`.

Alternatively, the `boot.flp` make target can be called, or the filesystem creating script, `src/release/scripts/doFS.sh`, may be invoked directly.

Local patches may also be supplied to the release build by defining the `LOCAL_PATCH` variable in `make release`.

### 5.2 Scripting `sysinstall`

The FreeBSD system installation and configuration tool, sysinstall(8), can be scripted to provide automated installs for large sites. This functionality can be used in conjunction with Intel® PXE[12] to bootstrap systems from the

network, or via custom boot floppies with a sysinstall script. An example sysinstall script is available in the CVS tree as `src/usr.sbin/sysinstall/install.cfg`.

# 6 Lessons Learned from FreeBSD 4.4

The release engineering process for 4.4 formally began on August 1st, 2001. After that date all commits to the `RELENG_4` branch of FreeBSD had to be explicitly approved by the Release Engineering Team <`re@FreeBSD.org`>. The first release candidate for the x86 architecture was released on August 16, followed by 4 more release candidates leading up to the final release on September 18th. The security officer was very involved in the last week of the process as several security issues were found in the earlier release candidates. A total of over *500* emails were sent to the Release Engineering Team <`re@FreeBSD.org`> in little over a month.

Our user community has made it very clear that the security and stability of a FreeBSD release should not be sacrificed for any self-imposed deadlines or target release dates. The FreeBSD Project has grown tremendously over its lifetime and the need for standardized release engineering procedures has never been more apparent. This will become even more important as FreeBSD is ported to new platforms.

# 7 Future Directions

It is imperative for our release engineering activities to scale with our growing userbase. Along these lines we are working very hard to document the procedures involved in producing FreeBSD releases.

- *Parallelism* - Certain portions of the release build are actually “embarrassingly parallel”. Most of the tasks are very I/O intensive, so having multiple high-speed disk drives is actually more important than using multiple processors in speeding up the `make release` process. If multiple disks are used for different hierarchies in the chroot(2) environment, then the CVS checkout of the `ports` and `doc` trees can be happening simultaneously as the `make world` on another disk. Using a RAID solution (hardware or software) can significantly decrease the overall build time.
- *Cross-building releases* - Building IA-64 or Alpha release on x86 hardware? `make TARGET=ia64 release`.
- *Regression Testing* - We need better automated correctness testing for FreeBSD.
- *Installation Tools* - Our installation program has long since outlived its intended life span. Several projects are under development to provide a more advanced installation mechanism. The libh project was one such project that aimed to provide an intelligent new package framework and GUI installation program.

# 8 Acknowledgements

I would like to thank Jordan Hubbard for giving me the opportunity to take on some of the release engineering responsibilities for FreeBSD 4.4 and also for all of his work throughout the years making FreeBSD what it is today. Of course the release would not have been possible without all of the release-related work done by Satoshi Asami <`asami@FreeBSD.org`>, Steve Price <`steve@FreeBSD.org`>, Bruce A. Mah <`bmah@FreeBSD.org`>, Nik Clayton <`nik@FreeBSD.org`>, David O’Brien <`obrien@FreeBSD.org`>, Kris Kennaway <`kris@FreeBSD.org`>, John Baldwin <`jhb@FreeBSD.org`> and the rest of the FreeBSD development community. I would also like to thank Rodney Grimes <`rgrimes@FreeBSD.org`>, Poul-Henning Kamp

<phk@FreeBSD.org>, and others who worked on the release engineering tools in the very early days of FreeBSD. This article was influenced by release engineering documents from the CSRG[13], the NetBSD Project[10], and John Baldwin's proposed release engineering process notes[11].

# 9 References

[1] CVS - Concurrent Versions System http://www.cvshome.org

[2] CVSup - The CVS-Optimized General Purpose Network File Distribution System http://www.polstra.com/projects/freeware/CVSup

[3] http://pointyhat.FreeBSD.org

[4] FreeBSD Ports Collection http://www.FreeBSD.org/ports

[5] FreeBSD Committers http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/contributors/staff-committers.html

[6] FreeBSD Core Team http://www.FreeBSD.org/administration.html#t-core

[7] FreeBSD Handbook http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook

[8] GNATS: The GNU Bug Tracking System http://www.gnu.org/software/gnats

[9] FreeBSD PR Statistics http://www.FreeBSD.org/prstats/index.html

[10] NetBSD Developer Documentation: Release Engineering http://www.NetBSD.org/developers/releng/index.html

[11] John Baldwin's FreeBSD Release Engineering Proposal http://people.FreeBSD.org/~jhb/docs/releng.txt

[12] PXE Jumpstart Guide http://www.FreeBSD.org/doc/en_US.ISO8859-1/articles/pxe/index.html

[13] Marshall Kirk McKusick, Michael J. Karels, and Keith Bostic: *The Release Engineering of 4.3BSD* (http://docs.FreeBSD.org/44doc/papers/releng.html)