Pondering Live CDs

Jan Schaumann

Abstract

Live CDs provide an easy and convenient way for users to try out a new operating system (OS) by letting them run the software directly from a bootable CD. While variations of such Live CDs exist for many free operating systems, they usually are developed for a specific purpose by third parties instead of being an integral and generic part of the release engineering process of the OS. Often marketed as a "free test-drive before switching", they generally do not actually accurately reflect what the operating system looks like after an initial installation, nor are they immediately available when a new version of the OS is released.

This paper investigates the concept of Live CDs and elaborates on the importance and marketing potential of the availability of a Live CD that's easy to install, use and expand. It also provides a simple conceptual framework for automatic generation of a generic NetBSD Live CD that may be integrated into NetBSD's release process.

1 Introduction

When it comes to Live CDs, most people think "Knoppix"! Well, actually, *most* people probabby think "What band?". However, in the special parallel universe inhabited by UNIX geeks of all shapes, the term "Live CD" used to be interchangeable with the Knoppix version of GNU/Linux. Tumbling deeper down the rabbit hole, in the corner occupied by the BSD camp, the name FreeSBIE has become fairly common. As the name suggests, FreeSBIE is a disc based on FreeBSD, but while most people in this BSD world have heard of FreeSBIE, not all that many people know that NetBSD has had Live CDs for a long time as well.

The first NetBSD based Live CD was based on NetBSD 1.5; since then, a number of Live CDs have been released, facilitated by a a small package in NetBSD's popular pkgsrc system that allows everybody to easily create their own Live CD (`pkgsrc/sysutils/mklivecd`[1]). Recently, OpenBSD based Live CDs have also been created.

All of these Live CDs[1] are of significant importance to all the BSD projects, as they provide an easy way of distributing a new release to users curious to test the operating system without having to actually install it. While other specialized CDs are available, this focus on new users remains one of the highest priorities, it seems, and a number of independent groups focus on creating such a CD, tweaking the setup, adding packages and providing a full desktop environment complete with browser, email client and office ware.

However, as useful as these products may be, they are not part of the actual operating system. Maintained by third parties, they do not actually do what they presumably set out to achieve: demonstrate what the OS looks like when a user chooses to install the release.

Instead, they provide an illusion, an ideal, an example of what the OS *could* look like, if you install it.[2] A normal installation from regular install media yields an entirely different operating system environment, and for a very good reason: the distributors of the OS do not claim to know what a user will want to use the OS for. They offer the user the largest amount of flexibility and let them tune the system to their specific needs.

In addition to locking a user into a pre-chosen and often

[1] The term "Live CD" may be a bit misleading, as the same concept applies to any kind of bootable media that provides a running operating system. This may include floppy discs, regular CDs, DVDs or bootable USB memory sticks and the like.

[2] Some CDs allow you to install the provided environment directly from the CD to your harddrive, which is a significant advantage; however, this approach still hides a number of complexities from the user.

desktop-centric environment, none of these products is actually integrated into the release engineering process of the respective operating system. This, of course, is a result of (a) being maintained by a third party and (b) the goal to add an ever increasing number of third-party applications, which can hardly be maintained by the OS release engineers.

Under NetBSD for example, the generation of a new release, including the various ISO images for the 53 different platforms for which binaries are created, is done from one central source tree using a complex `make(1)` framework. By adding a new "livecd" target, we could make sure that every new release automatically creates a bootable and runnable CD (for those architectures that support this) and provides users with the possibility to really test-drive the OS: instead of trying to impress users with complex graphical applications and a large number of finely tuned third-party application (all of which behave pretty much the same across operating system), this generic Live CD would offer a more solemn but honest "what you see is what you get" environment.

Even though not overly complex, such a Live CD would offer an amazing amount of flexibility: as it is indeed a full complete operating system, the user could take the CD and use it for almost all the different purposes for which they could use the regular OS. If done properly, adding the "livecd" `make` target would also allow users to create their own custom Live CDs even easier than previously.

2 Special Purpose Live CDs

The list of Live CDs available from the internet is impressive[2].[3] They range from large desktop oriented CDs that use compressed filesystems to add more and more applications to tiny little CDs that are used to set up small routers or firewalls in embedded environments. There is almost as wide a range of purposes for Live CDs as there is for the original OS itself. In addition, often there are specialized Live CDs that are used to automate the installation of the actual customized OS on a large scale.

In this section, we will take a look at some of these special purpose Live CDs, their requirements and consider how having a central framework to build a basic and generic Live CD can facilitate the expansion for any purpose users can dream up.

2.1 Install CDs

Even though it may not be immediately obvious, every BSD project already has at least one version of a Live CD: the regular install media. Install CDs (or floppies, a network boot setup, etc.) boot a version of the OS for a specific task: to install the OS. They provide a limited environment, but with a number of normal tools available. For that reason, install CDs can easily be used as so-called "rescue" CDs as well.

The installation process may differ, but in the end, the installer (whether it may be a curses based application or a complicated graphical program using a variety of toolkits) invokes the same programs as the user could manually.

Install CDs can, however, be also a lot more complex. Some versions of Unix offer media that allows for unattended installations (by retrieving the necessary information from the network at boot time), and System Administrators routinely cook up their own customized installation system that allows them to install a large set of machines without having to click "ok" a number of times on each machine.[4]

In order for a Live CD to function as an Install CD, it has the following requirements:

- a kernel with support for all required hardware: this is traditionally done via a so-called "GENERIC" kernel, that includes support for almost all devices available in the OS. Customized install CDs may, of course, only include the subset of drivers actually required with certain other options enabled as needed.
- tools to manipulate the target media: this commonly includes programs such as `fdisk(8)`, `disklabel(8)`, `installboot(8)`, `newfs(8)`, `mount(8)` etc.
- tools to retrieve the installation files: for network installations, this might include a DHCP client and `ftp(1)`; for non-network installations, the CD needs to provide the installation files as well
- tools to perform the actual installation: usually this includes `tar(1)` or `pax(1)`, `gzip(1)` or `bzip2(1)`, `mknod(8)`, etc. as well as, usually, a shell.

2.2 For the Desktop

This is the quintessential Live CD, targeted towards novice users who may wish to see a given OS in action before committing to installing it. The CD usually tries to automatically configure the X window system and comes pre-installed and -configured with a specific desktop environment. Since filesystem compression has become common, the developers of these CDs have been able to include more and more applications, providing full KDE or GNOME desktops in addition to full Office Suites, a browser, an email client, and even graphics manipulation programs, movie players and the like.

A regular user account is commonly provided in addition to the superuser account, allowing users to simply run any and all of the applications without having to install or configure

[3] Most of these happen to be Linux based, but that is actually tangential to the discussion.

[4] An example might be [14]; other ad-hoc clustering Live CDs such as [11], [12] and [13] would fall into the "Fixed Environment"category (Section 2.5).

anything. The CD attempts to simulate a full desktop, but since applications (and all the shared dependencies needed by them) need to be loaded from the CD into memory, the initial execution of any of these fairly large applications often makes them seem excruciatingly slow.

Desktop Live CDs like these are routinely given out for free to new prospective users at conferences, trade shows and user group meetings. Next to t-shirts (for which attendants seem willing to do just about anything), these CDs have proven to be particularly popular shwag. More recently, a growing number of Live CDs include the option of installing the environment that was loaded from the CD onto the harddrive, thus obsoleting the need for a separate installer. This allows users to not only test the environment, but to install it right away without having to worry about downloading and configuring the add-on applications they might otherwise need to obtain separately. While this is an appealing aspect of a Desktop oriented Live CD, it does bring with it a number of requirements:

- a kernel with support for all required hardware: again, this is traditionally done via the "GENERIC" kernel
- if the CD provides the ability to install the current environment, then it obviously needs to satisfy the requirements for an Install CD (see 2.1), and in addition the native package manager to maintain the add-on applications
- a utility to automatically detect the correct configuration and resolution of the X server
- an increasingly large number of complex third-party applications

This last requirement is what makes these Desktop Live CDs full-featured projects of their own: third-party applications are updated frequently, and with such updates come changes in their dependencies. In order to attract new users, the latest and greatest versions of all popular software need to be added, which makes it all but impossible to integrate the generation of such a CD into the proper release engineering process.

Especially for a portable OS like NetBSD, not all third-party applications are immediately available for all supported platforms, and the overhead of building and maintaining each application as part of the release process will quickly send your project's release engineers and system administrators running for the hills.

The most popular example of a desktop oriented Live CD is probably "Knoppix"[3]. BSD variations on this theme include "FreeSBIE"[4], "NewBIE"[5] and "OliveBSD"[6].

2.3 Swiss Army Knife

Targeted towards System Administrators, the typical "Swiss Army Knife" Live CD provides a bootable OS that includes all the tools necessary to troubleshoot system failures or network problems regardless of the installed OS. In practice, it often proves useful to not only include programs and utilities for the Live CD OS, but also to include native binaries for a range of other operating systems, private ssh keys, encryption and decryption tools, virus scanners and the like.

The general requirements for this type of Live CD are:

- a kernel suitable for the hardware in use at the site
- a number of third-party applications
- custom scripts, programs and files

In most cases, there is no need to run an X server from these Live CDs – a command-line only environment is usually fully sufficient.

While some of these Live CDs try to be the one tool for all purposes (in the true Swiss Army Knife spirit), they really start to shine once a System Administrator has customized it to his or her specific needs. A CD containing a number of native binaries for specific tools for all platforms that are supported, home-grown tools to connect to custom servers and automatically retrieve site-specific information, and a way to reinitialize or reinstall the target computer can save hours of troubleshooting work and minimize downtime drastically.

Obviously, it is difficult for anybody but the end-user to determine the optimal set of tools and files required. However, this does not mean that each user would need to reinvent the wheel and develop their own Live CD creation process from scratch. Once a simple infrastructure for creating a basic Live CD is in place, it becomes trivial to add the necessary applications and create a new, custom Live CD from these sources.

An examples of this kind of Live CD is "Frenzy"[7].

2.4 Focus on Security

Given the versatility of the BSD operating systems and their predominance in the server market, it comes as no supprise that several projects approach the concept of creating Live CDs from a security point of view. This includes a number of Live CDs that provide an embedded firewall or gateway, while some others may look like a variation of the Swiss Army Knife CD (see 2.3).

One of the things that make these Live CDs interesting is that most of them require a number of network services to be running and may need to log information or configuration changes to a writable location, which of course is not all that easy when running off read-only media. Routers/gateways and firewalls also often offer a web-interface to allow for runtime configuration, which implies running some kind of webserver from the Live CD, which initially may strike some as a somewhat unusual application for this environment.

The list of requirements for this type of Live CD is virtually identical to the ones for the Swiss Army Knife.

The most popular example for a Live CD with a focus on security and anonymity is probably "Anonym.OS"[8], a "bootable live cd based on OpenBSD that provides a hardened operating environment whereby all ingress traffic is denied and all egress traffic is automatically and transparently encrypted and/or anonymized."

2.5 Fixed Environment

Aside from general usage environments such as a desktop (even an anonymized and encrypted desktop), there is a number of scenarios that require an even more specific and fixed environment. A typical Kiosk setup for example is usually a very clearly defined setup with limited possibilities. A terminal allows anonymous users access to a specific resource, such as a library catalog, an interactive CD or a purchasing system, for example. Clearly, the user who uses the applications must not be allowed superuser access, and should even be prevented from accessing removable media, running any applications that are not part of the intended usage etc.

2.5.1 Example: Programming Contest

In 2004, Stevens Institute of Technology co-hosted the ACM Regional Collegiate Programming Contest for the Greater New York Region. In order to provide a consistent contest environment for the students at the local site as well as another remote location, a custom NetBSD Live CD[15] was developed, providing a well-defined and fixed multi-user environment. The CD contained all the compilers and debuggers, a number of common editors, the window manager and all standard tools as well as the specific contest software, mounting the home directories from a local NFS server.

This approach not only guaranteed equal contest conditions for teams at both locations, but also allowed for the restriction of network usage to a private subnet and prevented access to unauthorized devices while at the same time being unobtrusive enough to allow for a short setup and not require modifications to the existing workstations: all that needed to be done was to network the machines appropriately and boot off the CD.

3 Expandable Live CDs for everybody!

As we have seen in previous sections, the goals of the people creating and providing Live CDs vary widely. Almost all of the types of Live CDs mentioned above (with the obvious exception of the fixed environments) can be created in a generic way to appeal to the widest audience: one group of people creates an all-around desktop system, another group creates a CD with a collection of tools and programs that are useful for, say, digital forensics, and another group creates a CD that creates an instant firewall. However, all of these have certain disadvantages: they are either *too* general or do not provide enough flexibility.

A Live CD that allows the system to boot, discovers the network interface and configures it via DHCP, then drops the user into an installer may be useful for a small number of systems, but it is clearly far too generic to be deployed across hundred or thousands of machines. In order to automate the installation, the CD would need to be configured precisely for the given environment.

On the other hand, providing a selection of desktop applications – as impressive as the list may be – does not prove to be useful for more than a test-drive if certain other applications (or even preferred versions of some) are missing.

These problems are not very surprising, as each Live CD is developed with a very specific goal in mind (and providing a *broad* desktop is indeed a *specific* goal). But people often have very different ideas about how they would like to use a computer. Imagine if an operating system only came in one of a number of limited variations: you would have to choose between setting up a "desktop" or use a different OS (or a different installation of a given OS) if you wish to use some of the "Swiss Army Knife" tools!

After installing your operating system, you, the user, get to tune it and tweak it and fit it to your particular purpose. The BSD systems in general, and NetBSD in particular have refrained from trying to provide a "fully-configured" ready-for-all-purposes-out-of-the-box system; hundreds of third-party applications that may or may not be useful to the end user are *not* installed per default; network services are disabled and *not* every possible server is installed, set up and running. Instead, the user gets a full, complete operating system; a BSD system that allows the user to easily tune it to their specific needs, to add the applications required and to start the services desired, and nothing more. But nothing less, either.

It stands to reason that very often this is precisely what people would like to do with their Live CDs as well. So why not give all users the possibility to easily create their own Live CDs? Why not provide a Live CD of the system for every release that actually accurately reflects what the system looks like after installation?

3.1 Providing a Live CD for every release

As previously mentioned, virtually all Live CD are maintained by third parties. That is, the creation of a Live CD is not done by the same people who create the operating system, and is not part of the release process. That means that every new release makes the existing Live CDs outdated, and the third party has to catch up and create a new version of their product.

Considering how often Live CDs are used as a free giveaway to attract users to the operating system, this seems quite surprising. Since Live CDs *are* an important tool in attracting new users and *are* used to test the operating system, the

importance of automatically providing a Live CD with every release should be obvious.

So far, this has not happened. While NetBSD's release engineering process automatically generates bootable install CDs for every architecture, these install CDs are very limited and do not offer the full-featured operating system that users could sample. The addition of a new "livecd" target as discussed earlier on (section 1) is not overly complex and can integrated fairly easily.[5] This way, the NetBSD Project could provide an up-to-date general-purpose Live CD for each release, while at the same time providing a simple framework that allows users to extend this basic Live CD just like they have the option of extending the default installation.

3.2 Extending a simple Live CD

Once a framework is in place to create a general purpose Live CD, it will be trivial to develop custom Live CDs. When creating the general purpose Live CD, we take advantage of the fact that during the build of a NetBSD release a destination directory is already populated with the operating system, from which binary sets are then created. Mounting the release directory into the destination directory and building the `sysinst` tool here as well then yields a Live CD that allows the installation of the operating system without any added complexity.

After building a release, the user can then populate the destination directory with whatever applications she may need, perform whatever changes she sees fit and in general tweak and tune the system to her heart's desire. After all this is done, a simple `make livecd` will create the desired CD.

Please see appendix B for further notes on the provided sample implementation of the "livecd" target.

4 Conclusion

When I set out to write this paper, my intention was to focus on NetBSD Live CDs. However, obviously the same concepts apply to other operating systems as well. While the findings of this paper may not be particularly groundbreaking or technically complex, through discussions with other developers from a number of projects, I've found that the importance of having even only a basic, all-purpose Live CD for every new release is not to be underestimated. With this analysis, I'd like to encourage all of the BSD projects to make it a priority to allow users the flexibility to (a) try a new operating system and actually see exactly what it looks like and (b) to rely on the operating system's standard framework to provide an easy to use basis to build their own Live CDs.

The efforts that the existing Live CD projects put into their work remains valuable and appreciated, and an "official" Live CD provided by the project developers will only add to the already impressive list of uses we can put our favorite operating system to.

[5] A proof of concept implementation for the i386 platform – by far the most popular platform for Live CDs at the moment – can be found in Appendix A.1.

5 Author Information

Jan Schaumann is a System Administrator and NetBSD developer. He holds an MS in Computer Science from Stevens Institute of Technology, where he also works, performing a sometimes surprising range of duties. Jan lives in New York City together with his wife and will happily drop anything (including his laptop!) in order to go skateboarding, snowboarding and surfing. You can reach him at $< jschauma@netmeister.org >$.

A Code Listings

A.1 Proof of concept implementation of a “livecd” target

```
Index: Makefile
===================================================================
RCS file: /cvsroot/src/etc/Makefile,v
retrieving revision 1.324
diff -b -u -r1.324 Makefile
--- Makefile 17 Feb 2006 22:09:33 -0000 1.324
+++ Makefile 19 Apr 2006 02:28:24 -0000
@@ -65,6 +65,11 @@
 MKISOFS_FLAGS+= -quiet
 .endif

+# same as above, for makefs
+MAKEFS?= makefs
+MAKEFS_FLAGS+= -t cd9660 \
+ -o "allow-max-name,rockridge,label=\"The NetBSD Project\""
+

 # MD Makefile.inc may append MD targets to BIN[123].  Make sure all
 # are empty, to preserve the old semantics of setting them below with "+=".
@@ -393,6 +398,62 @@
  ${KERNEL_SETS:@.SETS.@kern-${.SETS.}.tgz@}
  ${MAKESUMS} -t ${RELEASEDIR}/${RELEASEMACHINEDIR}/binary/kernel '*.gz'

+# livecd --
+# Standalone target to create a live CDROM image after the release
+# was composed.  Should be run after "make release" in src and xsrc.
+#
+#LIVE.image=${RELEASEDIR}/${RELEASEMACHINEDIR}/installation/cdrom/netbsd-${MACHINE}${CDR
+LIVE.image=${BSDOBJDIR}/netbsd-${MACHINE}${CDROM_NAME_ADD}-live.iso
+
+.if "${MACHINE_ARCH}" == "i386"
+
+# XXX
+echo:
+ @echo ${BSDOBJDIR}
+
+livecd: .PHONY check_DESTDIR check_RELEASEDIR livecd-setup
+ ${MAKEFS} ${MAKEFS_FLAGS} ${LIVE.image} ${DESTDIR}
+
+livecd-clean: .PHONY check_DESTDIR
+ umount ${DESTDIR}/${MACHINE_ARCH}
+ rmdir ${DESTDIR}/${MACHINE_ARCH}
+ rmdir ${DESTDIR}/.etcrw ${DESTDIR}/.varrw ${DESTDIR}/proc
+ rm -f ${DESTDIR}/cdboot ${DESTDIR}/boot ${DESTDIR}/etc/fstab
+ rm -f ${DESTDIR}/etc/rc.conf ${DESTDIR}/etc/rc.d/livecd
+ rm -fr ${DESTDIR}/usr/local
+
+livecd-setup: .PHONY check_RELEASEDIR mount-union-release livecd-sysinst livecd-cdboot
+ mkdir -p ${DESTDIR}/.etcrw ${DESTDIR}/.varrw ${DESTDIR}/proc
+ install -c -m 444 -o root -g wheel  \
```

```
+ ${NETBSDSRCDIR}/etc/etc.${MACHINE_ARCH}/fstab.live  \
+ ${DESTDIR}/etc/fstab
+ install -c -m 444 -o root -g wheel  \
+ ${NETBSDSRCDIR}/etc/etc.${MACHINE_ARCH}/rc.conf.live  \
+ ${DESTDIR}/etc/rc.conf
+ install -c -m 755 -o root -g wheel  \
+ ${NETBSDSRCDIR}/etc/rc.d/livecd  \
+ ${DESTDIR}/etc/rc.d/livecd
+ install -c -d ${DESTDIR}/usr/local/sbin
+ install -c -m 755 -o root -g wheel  \
+ ${NETBSDSRCDIR}/distrib/utils/sysinst/arch/${MACHINE_ARCH}/obj/sysinst \
+ ${DESTDIR}/usr/local/sbin/sysinst
+ install -c -m 755 -o root -g wheel  \
+ ${NETBSDSRCDIR}/sys/arch/${MACHINE_ARCH}/stand/cdboot/cdboot \
+ ${DESTDIR}/cdboot
+ install -c -m 444 -o root -g wheel \
+ ${DESTDIR}/usr/mdec/boot ${DESTDIR}/boot
+
+mount-union-release: .PHONY
+ mkdir -p ${DESTDIR}/${MACHINE_ARCH}
+ mount_null ${RELEASEDIR}/${MACHINE_ARCH} ${DESTDIR}/${MACHINE_ARCH}
+
+livecd-cdboot: .PHONY
+ ( cd ${NETBSDSRCDIR}/sys/arch/${MACHINE_ARCH}/stand/cdboot && make dependall )
+
+livecd-sysinst: .PHONY
+ ( cd ${NETBSDSRCDIR}/distrib/utils/sysinst && make dependall )
+
+.endif

 # iso-image --
 # Standalone target to create a CDROM image after the release
Index: etc.i386/Makefile.inc
===================================================================
RCS file: /cvsroot/src/etc/etc.i386/Makefile.inc,v
retrieving revision 1.41
diff -b -u -r1.41 Makefile.inc
--- etc.i386/Makefile.inc 11 Mar 2005 20:55:10 -0000 1.41
+++ etc.i386/Makefile.inc 19 Apr 2006 02:28:24 -0000
@@ -25,3 +25,5 @@
 CDROM_BOOT_IMAGE?= boot-big.fs

 MKISOFS_FLAGS+= -b ${MACHINE}/installation/floppy/${CDROM_BOOT_IMAGE} -c boot.catalog
+
+MAKEFS_FLAGS+= -o "bootimage=i386;${DESTDIR}/cdboot"
--- etc.i386/fstab.live 2006-04-08 14:39:41.000000000 -0400
+++ /dev/null 2006-04-18 22:21:09.000000000 -0400
@@ -1,4 +0,0 @@
-/ignoreme       /tmp    mfs     rw,-b4096,-f512,-s262144        0 0
-/.etcrw /etc    union   rw                                      0 0
-/.varrw /var    union   rw                                      0 0
-procfs          /proc   procfs  rw
--- etc.i386/rc.conf.live 2006-04-08 14:49:43.000000000 -0400
+++ /dev/null 2006-04-18 22:21:09.000000000 -0400
```

```
@@ -1,30 +0,0 @@
-# $NetBSD: rc.conf,v 1.96 2000/10/14 17:01:29 wiz Exp $
-#
-# see rc.conf(5) for more information.
-#
-# Use program=YES to enable program, NO to disable it. program_flags are
-# passed to the program on the command line.
-#
-
-# Load the defaults in from /etc/defaults/rc.conf (if it’s readable).
-# These can be overridden below.
-#
-if [ -r /etc/defaults/rc.conf ]; then
-. /etc/defaults/rc.conf
-fi
-
-# If this is not set to YES, the system will drop into single-user mode.
-#
-rc_configured=YES
-
-# Add local overrides below
-#
-livecd=YES
-no_swap=YES
-sendmail=NO
-syslogd=NO
-inetd=NO
-savecore=NO
-
-hostname=livecd
-#dhclient=yes dhclient_flags=-q
--- rc.d/livecd 2006-04-18 22:25:11.000000000 -0400
+++ /dev/null 2006-04-18 22:21:09.000000000 -0400
@@ -1,29 +0,0 @@
-#!/bin/sh
-#
-# $NetBSD: $
-#
-
-# PROVIDE: livecd
-# BEFORE: disks
-
-$_rc_subr_loaded . /etc/rc.subr
-
-name="livecd"
-start_cmd="livecd_start"
-stop_cmd=":"
-
-livecd_start()
-{
-(
-umask 022
-mount_mfs -b4096 -f512 -s5m swap /.etcrw
-mount_mfs -b4096 -f512 -s10m swap /.varrw
```

```
-chmod 755 /.etcrw /.varrw
-mount_union /.etcrw /etc
-mount_union /.varrw /var
-cp /etc/motd /.etcrw
-)
-}
-
-load_rc_config $name
-run_rc_command "$1"
```

B Notes on the sample implementation

The proof of concept implementation of a “livecd” target for the i386 platform does require a number of improvements before it can be integrated completely. The following are a few notes of open issues:

B.1 “makefs” vs. “mkisofs”

The sample implementation uses the `makefs(8)` tool to create the ISO 9660 CD filesystem instead of relying on the third-party application `mkisofs`. This has the advantage of not requiring a separate tool that needs to be installed form pkgsrc. However, since the current implementation of the “iso-image” target still relies on `mkisofs`, there is not much gained at the moment.

B.2 “cdboot” vs. floppy/hard disk emulation

The primary bootloader “cdboot” was added in June 2005 by Bang Jun-Young. This bootloader loads a secondary boot loader directly from CD without performing floppy/hard disk emulation as described by the El Torito specification. It is used to load the CD, but is, at the moment, only available for the i386 and amd64 ports. As Live CDs are created for other ports, the target may need to be adapted to use the old floppy emulation.

B.3 Writing data

As the entire operating system is loaded from read-only media, the user can not write any data. To overcome this, the “livecd” rc script creates a couple of memory mounted filesystems. It would be great if this behaviour could be influenced by some variable that the user can set when building the Live CD.

B.4 Compressed filesystem

The generic Live CD creates by this approach does not use a compressed filesystem. That means, it only allows for a very limited amount of add-on applications. This can easily be overcome by using `vndcompress(1)` to compress the appropriate portions of the filesystem and uncompressing it at boot time.

References

[1] *mklivecd* Juan Romero Pardines, `pkgsrc/sysutils/mklivecd`

[2] `http://www.frozentech.com/content/livecd.php`

[3] *KNOPPIX* Klaus Knopper, `http://www.knoppix.org`

[4] *FreeSBIE* Davide d'Amico, Dario Freni et al, `http://www.freesbie.org`

[5] *NewBIE* TDI Security, `http://arudius.sourceforge.net/`

[6] *OliveBSD* Gabriel Paderni, `http://g.paderni.free.fr/olivebsd/`

[7] *Project Frenzy* `http://frenzy.org.ua/eng/`

[8] *Anonym.OS* L. Taylor Banks, `http://sourceforge.net/projects/anonym-os/`

[9] *XORP* Open Source IP Router, `http://www.xorp.org/livecd.html`

[10] *STD* Security Tools Distribution, `http://s-t-d.org/`

[11] *Clusterix* `http://clusterix.livecd.net/`

[12] *ClusterKnoppix* `http://clusterknoppix.sw.be/`

[13] *Parallel Knoppix* `http://pareto.uab.es/mcreel/ParallelKnoppix/`

[14] *Warewulf* `http://www.warewulf-cluster.org/cgi-bin/trac.cgi`

[15] *NetBSD/ACM Bootable CD* Jan Schaumann, `http://www.cs.stevens.edu/~jschauma/acm/`