mfsBSD

Toolset to create memory filesystem based FreeBSD distributions

Martin Matuška
mm@FreeBSD.org

May 26, 2009

Abstract

mfsBSD is a toolset to create small-sized but full-featured mfsroot based distributions of FreeBSD that store all files in memory (MFS) and load from hard drive, usb storage device or optical media. It can be used for a variety of purposes, including diskless systems, recovery partitions and remotely overwriting other operating systems. This paper describes design, structure and usage of mfsBSD.

1 History

mfsBSD is originally based on ideas and code parts from a script set called depenguinator [3]. The goal of depenguinator was to allow installing FreeBSD on dedicated servers that only offer Linux distributions as an operating system option. After rebooting a FreeBSD system was fully loaded into memory and the user could partition and format disk drives and install FreeBSD on these servers. The original depenguinator was created in december 2003 for use with FreeBSD 5.x. This version was incompatible with FreeBSD 6.x and 7.x. An updated Depenguinator 2.0 was released in January, 2008. Depenguinator is designed to create FreeBSD disk images on Linux computers and uses makefs from NetBSD.

The mfsBSD development started in late 2007 by Martin Matuška (mm@FreeBSD.org), trying to provide depenguinator functionality for FreeBSD 6.x. The approach was significantly simplified with the new features in the rcng startup scripts and in addition, the geom_uzip class was used to compress the /usr filesystem and save space. mfsBSD requires FreeBSD to create images and compared to depenguinator it can create ISO images and tar-gz compressed filesystems as well.

In April 2008, Daniel Gerzo (danger@FreeBSD.org) published an article called "Remote Installation of the FreeBSD Operating System without a Remote Console"[1] that is now a part of the FreeBSD documentation collection . The setup in this article uses mfsBSD for installation of remote systems.

2 Use cases

mfsBSD may be deployed in various scenarios. The following list of examples shows where mfsBSD might be useful:

- Remote Install over Network
- Live-System (USB-Stick or optical media)
- Rescue Partition
- FreeBSD Upgrade

The scenarios above are described in the following subsections.

2.1 Remote Install over Network

mfsBSD can be used to owerwrite a remote installation by copying the raw image on the beginning of the system's primary drive. This is reported to by successful on systems with various Linux distributions and is suitable for use with dedicated server providers that neighter offer FreeBSD nor provide a remote drive and/or remote console function.

2.2 Live-System

Another use of mfsBSD is a live-system installation suitable for an USB-Stick or optical media. It provides the FreeBSD base system and some user-supplied packages, but this is limited by the maximal image size of mfsroot which is around 50MB. This system may be used for a FreeBSD installation, as a rescue media to access a system that lost it's ability to boot properly or as an embeeded FreeBSD installation (firewall, etc.).

2.3 Rescue-Partition

A mfsBSD image fits on a small-sized 64MB partition that may be used for rescue system purposes. This partition cat be booted into from the FreeBSD boot loader (or any other boot loader that supports FreeBSD partitions). With a system fully loaded into memory, user may modify hard drive partitions, boot sectors, etc.

2.4 FreeBSD Upgrade

FreeBSD can be easily upgraded to a new major version or switch architecture between i386 and amd64 using mfsBSD. The boot part of an existing system can be modified to boot using mfsBSD in a very few steps. These are described in section 5 of this document.

3 Technical Background

mfsBSD consists of a BSD Makefile, rcNG scripts, configuration fules and text documentation files. The operation of the mfsBSD toolset is very simple and follows the following steps:

1. Prepare and deploy kernel and base files
   Kernel and base files may be extracted from a mounted FreeBSD CD-ROM (which may be a md-mounted ISO image) or by using make installkernel and installworld from FreeBSD sources.

2. Remove unnecessary files
   Unnecessary files like manual pages and documentation are removed.figuration files and custom scripts

3. Process and deploy configuration files
   Required and optional configuration files are installed and custom startup scripts are added to the target image. The only required modification to the FreeBSD rcNG is addition of the **mdinit** script that mounts the compressed filesystem on boot.

4. Compress /usr filesystem with uzip
   The /usr directory is compressed with mkuzip and accessed via geom_uzip.

5. Add user packages
   Users may select custom FreeBSD packages for inclusion in the image. The creation of /usr/local and deployment of the packages is managed in the **packages** script.

6. Build mfsroot
   The mfsroot image is built.

7. Create deployable output image
   raw (disk file), ISO or a tar.gz image is created.

Variable	Default	Description
BASE	/cdrom/7.X-RELEASE	points to base installation files
IMAGE	mfsboot.img	name of the resulting raw image
ISOIMAGE	mfsboot.iso	name of the resulting ISO image
TARFILE	mfsboot.tar.gz	name of the resulting tar.gz image
KERNCONF	mfsboot.tar.gz	kernel configuration file
CUSTOM	*undefined*	install custom kernel and world
BUILDWORLD	*undefined*	build and install custom world (needs CUSTOM)
BUILDKERNEL	*undefined*	build and install custom kernel (needs CUSTOM)

Table 1: User-defined variables of mfsBSD Makefile

The uzip and mfsroot umages are created using the doFS.sh[1] script from FreeBSD.

4 Image Types

The Makefile of mfsBSD suppports the following main targets:

- image - create raw image file, default (this image can be directly streamed to a drive)
- iso - create a bootable ISO image
- tar - create tar.gz with kernal and mfsroot that can be uncompressed on a FreeBSD partition

mfsBSD uses files from the FreeBSD CD installation by default. Makefile supports several options, some of these are presented in table 1.

5 Usage tutorial

This section contains a short tutorial of building a simple FreeBSD image using mfsBSD tools.

1. Download and extract mfsBSD tools from the Homepage [2]
   `tar xfz mfsbsd-XXXX.tar.gz`

2. Download and mount CD 1 (or ISO file) of a FreeBSD release
   `mdconfig -a -t vnode -f 7.2-RELEASE-amd64.ISO`
   `mount_cd9660 /dev/mdX /cdrom`

3. Modify configuration files in *conf/* to your needs
   - setup network interfaces and default gateway (*conf/rc.conf*)
   - root password (*conf/rootpw.conf* - optional)
   - ssh keys (*conf/authorized_keys* - optional)
   - nameservers (*conf/resolv.conf*)

4. Build an image
   `make` - build raw image
   `make tar` - build tar.gz file
   `make iso` - build ISO image

5. Deploy image to target media
   Now you can deploy the image to a storage device (hard drive, USB-drive or optical media) or to a FreeBSD partition (tar.gz).

[1] doFS.sh is available from http://www.freebsd.org/cgi/cvsweb.cgi/src/release/scripts/doFS.sh

References

[1] Daniel Gerzo. Remote installation of the freebsd operating system without a remote console, April 2008. URL http://www.freebsd.org/doc/en/articles/remote-install/.

[2] Martin Matuska. mfsBSD Homepage, 2009. URL http://people.freebsd.org/~mm/mfsbsd/.

[3] Colin Percival. Depenguinator, December 2003. URL http://www.daemonology.net/depenguinator/.