Remote Installation of the FreeBSD Operating System Without a Remote Console

Daniel Gerzo <danger@FreeBSD.org >

Revision: 45156

Copyright © 2008 The FreeBSD Documentation Project

FreeBSD is a registered trademark of the FreeBSD Foundation.

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.

2014-06-29 by eadler.

Abstract

This article documents the remote installation of the FreeBSD operating system when the console of the remote system is unavailable. The main idea behind this article is the result of a collaboration with Martin Matuska <mm@FreeBSD.org > with valuable input provided by Pawel Jakub Dawidek <pjd@FreeBSD.org >.

Table of Contents

1. Background 1
2. Introduction 2
3. Preparation - mfsBSD 2
4. Installation of the FreeBSD Operating System 5
5. ZFS 8

1. Background

There are many server hosting providers in the world, but very few of them are officially supporting FreeBSD. They usually provide support for a Linux® distribution to be installed on the servers they offer.

In some cases, these companies will install your preferred Linux® distribution if you request it. Using this option, we will attempt to install FreeBSD. In other cases, they may offer a rescue system which would be used in an emergency. It is possible to use this for our purposes as well.

This article covers the basic installation and configuration steps required to bootstrap a remote installation of FreeBSD with RAID-1 and ZFS capabilities.

2. Introduction

This section will summarize the purpose of this article and better explain what is covered herein. The instructions included in this article will benefit those using services provided by colocation facilities not supporting FreeBSD.

1. As we have mentioned in the Background section, many of the reputable server hosting companies provide some kind of rescue system, which is booted from their LAN and accessible over SSH. They usually provide this support in order to help their customers fix broken operating systems. As this article will explain, it is possible to install FreeBSD with the help of these rescue systems.

2. The next section of this article will describe how to configure, and build minimalistic FreeBSD on the local machine. That version will eventually be running on the remote machine from a ramdisk, which will allow us to install a complete FreeBSD operating system from an FTP mirror using the sysinstall utility.

3. The rest of this article will describe the installation procedure itself, as well as the configuration of the ZFS file system.

2.1. Requirements

To continue successfully, you must:

- Have a network accessible operating system with SSH access
- Understand the FreeBSD installation process
- Be familiar with the sysinstall(8) utility
- Have the FreeBSD installation ISO image or CD handy

3. Preparation - mfsBSD

Before FreeBSD may be installed on the target system, it is necessary to build the minimal FreeBSD operating system image which will boot from the hard drive. This way the new

system can be accessed from the network, and the rest of the installation can be done without remote access to the system console.

The mfsBSD tool-set can be used to build a tiny FreeBSD image. As the name of mfsBSD suggests ("mfs" means "memory file system"), the resulting image runs entirely from a ramdisk. Thanks to this feature, the manipulation of hard drives will not be limited, therefore it will be possible to install a complete FreeBSD operating system. The mfsBSD `home page` includes pointers to the latest release of the toolset.

Please note that the internals of mfsBSD and how it all fits together is beyond the scope of this article. The interested reader should consult the original documentation of mfsBSD for more details.

Download and extract the latest mfsBSD release and change your working directory to the directory where the mfsBSD scripts will reside:

```
# fetch http://people.freebsd.org/~mm/mfsbsd/mfsbsd-latest.tar.gz
# tar xvzf mfsbsd-1.0-beta1.tar.gz
# cd mfsbsd-1.0-beta1/
```

3.1. Configuration of mfsBSD

Before booting mfsBSD, a few important configuration options have to be set. The most important that we have to get right is, naturally, the network setup. The most suitable method to configure networking options depends on whether we know beforehand the type of the network interface we will use, and the network interface driver to be loaded for our hardware. We will see how mfsBSD can be configured in either case.

Another important thing to set is the `root` password. This can be done by editing `conf/rootpw.conf`. Please keep in mind that the file will contain your password in the plain text, thus we do not recommend to use real password here. Nevertheless, this is just a temporary one-time password which can be later changed in a live system.

3.1.1. The `conf/interfaces.conf` method

When the installed network interface card is unknown, we can use the auto-detection features of mfsBSD. The startup scripts of mfsBSD can detect the correct driver to use, based on the MAC address of the interface, if we set the following options in `conf/interfaces.conf`:

```
initconf_interfaces="ext1"
initconf_mac_ext1="00:00:00:00:00:00"
initconf_ip_ext1="192.168.0.2"
initconf_netmask_ext1="255.255.255.0"
```

Do not forget to add the `defaultrouter` information to `conf/rc.conf`:

```
defaultrouter="192.168.0.1"
```

3.1.2. The `conf/rc.conf` Method

When the network interface driver is known, it is more convenient to use `conf/rc.conf` for networking options. The syntax of this file is the same as the one used in the standard rc.conf(5) file of FreeBSD.

For example, if you know that a re(4) network interface is going to be available, you can set the following options in `conf/rc.conf` :

```
defaultrouter="192.168.0.1"
ifconfig_re0="inet 192.168.0.2 netmask 255.255.255.0"
```

3.2. Building an mfsBSD Image

The process of building an mfsBSD image is pretty straightforward.

The first step is to mount the FreeBSD installation CD, or the installation ISO image to `/cdrom`. For the sake of example, in this article we will assume that you have downloaded the FreeBSD 7.0-RELEASE ISO. Mounting this ISO image to the `/cdrom` directory is easy with the mdconfig(8) utility:

```
# mdconfig -a -t vnode -u 10 -f  7.0-RELEASE-amd64-disc1.iso
# mount_cd9660 /dev/md10 /cdrom
```

Next, build the bootable mfsBSD image:

```
# make BASE=/cdrom/ 7.0-RELEASE
```

Note

The above `make` has to be run from the top level of the mfsBSD directory tree, for example `~/mfsbsd-1.0-beta1/`.

3.3. Booting mfsBSD

Now that the mfsBSD image is ready, it must be uploaded to the remote system running a live rescue system or pre-installed Linux® distribution. The most suitable tool for this task is scp:

```
# scp disk.img root@192.168.0.2:.
```

To boot mfsBSD image properly, it must be placed on the first (bootable) device of the given machine. This may be accomplished using this example providing that `sda` is the first bootable disk device:

```
# dd if=/root/disk.img of=/dev/sda bs=1m
```

If all went well, the image should now be in the MBR of the first device and the machine can be rebooted. Watch for the machine to boot up properly with the ping(8) tool. Once it has came back on-line, it should be possible to access it over ssh(1) as user `root` with the configured password.

4. Installation of the FreeBSD Operating System

The mfsBSD has been successfully booted and it should be possible to log in through ssh(1). This section will describe how to create and label slices, set up gmirror for RAID-1, and how to use sysinstall to install a minimal distribution of the FreeBSD operating system.

4.1. Preparation of Hard Drives

The first task is to allocate disk space for FreeBSD, i.e.: to create slices and partitions. Obviously, the currently running system is fully loaded in system memory and therefore there will be no problems with manipulating hard drives. To complete this task, it is possible to use either sysinstall or fdisk(8) in conjunction to bsdlabel(8).

At the start, mark all system disks as empty. Repeat the following command for each hard drive:

```
# dd if=/dev/zero of=/dev/ ad0 count=2
```

Next, create slices and label them with your preferred tool. While it is considered easier to use sysinstall, a powerful and also probably less buggy method will be to use standard text-based UNIX® tools, such as fdisk(8) and bsdlabel(8), which will also be covered in this section. The former option is well documented in the Installing FreeBSD chapter of the FreeBSD Handbook. As it was mentioned in the introduction, this article will present how to set up a system with RAID-1 and ZFS capabilities. Our set up will consist of a small gmirror(8) mirrored `/` (root), `/usr` and `/var` dataset, and the rest of the disk space will be allocated for a zpool(8) mirrored ZFS file system. Please note, that the ZFS file system will be configured after the FreeBSD operating system is successfully installed and booted.

The following example will describe how to create slices and labels, initialize gmirror(8) on each partition and how to create a UFS2 file system in each mirrored partition:

```
# fdisk -BI /dev/ad0  ❶
# fdisk -BI /dev/ad1
# bsdlabel -wB /dev/ad0s1  ❷
# bsdlabel -wB /dev/ad1s1
# bsdlabel -e /dev/ad0s1  ❸
# bsdlabel /dev/ad0s1 > /tmp/bsdlabel.txt && bsdlabel -R /dev/ad1s1 /
tmp/bsdlabel.txt  ❹
# gmirror label root /dev/ad[01]s1a  ❺
# gmirror label var /dev/ad[01]s1d
# gmirror label usr /dev/ad[01]s1e
```

```
# gmirror label -F swap /dev/ad[01]s1b   ❻
# newfs /dev/mirror/root   ❼
# newfs /dev/mirror/var
# newfs /dev/mirror/usr
```

❶ Create a slice covering the entire disk and initialize the boot code contained in sector 0 of the given disk. Repeat this command for all hard drives in the system.
❷ Write a standard label for each disk including the bootstrap code.
❸ Now, manually edit the label of the given disk. Refer to the bsdlabel(8) manual page in order to find out how to create partitions. Create partitions `a` for `/` (root) file system, `b` for swap, `d` for `/var`, `e` for `/usr` and finally `f` which will later be used for ZFS.
❹ Import the recently created label for the second hard drive, so both hard drives will be labeled in the same way.
❺ Initialize gmirror(8) on each partition.
❻ Note that `-F` is used for the swap partition. This instructs gmirror(8) to assume that the device is in the consistent state after the power/system failure.
❼ Create a UFS2 file system on each mirrored partition.

4.2. System Installation

This is the most important part. This section will describe how to actually install the minimal distribution of FreeBSD on the hard drives that we have prepared in the previous section. To accomplish this goal, all file systems need to be mounted so sysinstall may write the contents of FreeBSD to the hard drives:

```
# mount /dev/mirror/root /mnt
# mkdir /mnt/var /mnt/usr
# mount /dev/mirror/var /mnt/var
# mount /dev/mirror/usr /mnt/usr
```

When you are done, start sysinstall(8). Select the Custom installation from the main menu. Select Options and press Enter. With the help of arrow keys, move the cursor on the `Install Root` item, press Space and change it to `/mnt`. Press Enter to submit your changes and exit the Options menu by pressing q.

Warning

Note that this step is very important and if skipped, sysinstall will be unable to install FreeBSD.

Go to the Distributions menu, move the cursor with the arrow keys to `Minimal`, and check it by pressing Space. This article uses the Minimal distribution in order to save network traffic, because the system itself will be installed over ftp. Exit this menu by choosing `Exit`.

Note

The Partition and Label menus will be skipped, as these are useless now.

In the Media menu, select FTP. Select the nearest mirror and let sysinstall assume that the network is already configured. You will be returned back to the Custom menu.

Finally, perform the system installation by selecting the last option, Commit. Exit sysinstall when it finishes the installation.

4.3. Post Installation Steps

The FreeBSD operating system should be installed now; however, the process is not finished yet. It is necessary to perform some post installation steps in order to allow FreeBSD to boot in the future and to be able to log in to the system.

You must now chroot(8) into the freshly installed system in order to finish the installation. Use the following command:

```
# chroot /mnt
```

To complete our goal, perform these steps:

- Copy the GENERIC kernel to the /boot/kernel directory:

  ```
  # cp -Rp /boot/GENERIC/* /boot/kernel
  ```

- Create the /etc/rc.conf , /etc/resolv.conf and /etc/fstab files. Do not forget to properly set the network information and to enable sshd in /etc/rc.conf . The contents of /etc/fstab will be similar to the following:

  ```
  # Device                Mountpoint      FStype  Options         Dump
      Pass#
  /dev/mirror/swap        none            swap    sw              0
       0
  /dev/mirror/root        /               ufs     rw              1
       1
  /dev/mirror/usr         /usr            ufs     rw              2
       2
  /dev/mirror/var         /var            ufs     rw              2
       2
  /dev/cd0                /cdrom          cd9660  ro,noauto       0
       0
  ```

- Create /boot/loader.conf with the following contents:

```
geom_mirror_load="YES"
zfs_load="YES"
```

- Perform the following command, which will make ZFS available on the next boot:

```
# echo 'zfs_enable="YES"' >> /etc/rc.conf
```

- Add additional users to the system using the adduser(8) tool. Do not forget to add a user to the `wheel` group so you may obtain root access after the reboot.
- Double-check all your settings.

The system should now be ready for the next boot. Use the reboot(8) command to reboot your system.

5. ZFS

If your system survived the reboot, it should now be possible to log in. Welcome to the fresh FreeBSD installation, performed remotely without the use of a remote console!

The only remaining step is to configure zpool(8) and create some zfs(8) file systems. Creating and administering ZFS is very straightforward. First, create a mirrored pool:

```
# zpool create tank mirror /dev/ad[01]s1f
```

Next, create some file systems:

```
# zfs create tank/ports
# zfs create tank/src
# zfs set compression=gzip tank/ports
# zfs set compression=on tank/src
# zfs set mountpoint=/usr/ports tank/ports
# zfs set mountpoint=/usr/src tank/src
```

That is all. If you are interested in more details about ZFS on FreeBSD, please refer to the ZFS section of the FreeBSD Wiki.