FreeBSD Experience

new This page is about my experience using FreeBSD.

Getting FreeBSD

Installer images are here. It is recommended to get the latest, which is, currently, the FreeBSD 14.2-RELEASE. Both ISO (DVD) and memstick (USB@solid-state) images are available.

Note that FreeBSD uses the name `amd64` to refer to the x86_64 architecture. So, most PC users would get either the FreeBSD-14.2-RELEASE-amd64-dvd1.iso (~5GB@~4.5GiB: huge, all-in) or the FreeBSD-14.2-RELEASE-amd64-memstick.img (~1.5GB: slightly smaller, may need internet connection while installing). Smaller installers are available as FreeBSD-14.2-RELEASE-amd64-bootonly.iso (~460MB) and FreeBSD-14.2-RELEASE-amd64-memstick.img (~560MB), but would most probably need an internet connect during installation.

2026/02/09 22:23

Installing FreeBSD

Latest: Installing FreeBSD-14.2-RELEASE-amd64 (**202501070736** Use dd to write the image to a USB drive)

Base System

A fresh install - full hard disk available.

- boot installer
- select lib32 (kernel and base always selected)
- added main user account to {*wheel*,*operator*} group
  - for desktop use, all users need to be in *video* or *wheel*
  - so normal users should be in *video* group
  - normal users should also be in *operator* group (some DE need this)
- enable sshd, enable ntpdate(on-boot), disable sendmail

Desktop (GUI)

Information from this book.

Lightweight path... using `pkg` to install

- going jwm

```
# pkg install -y xorg jwm xlockmore xfontsel
```

    - a more newbie-friendly option could be

```
# pkg install -y xorg mate libreoffice firefox
```

    - also

```
# pkg install -y lightdm lightdm-gtk-greeter
```

- needed if group not assigned during user creation

```
# pw groupmod video -m user || pw groupmod wheel -m user
```

- basic X should work (as user)

```
$ echo "exec jwm" >.xinitc
$ startx
```

- get proper graphics driver (list from handbook)
    - e.g. intel/drm-kmod/i915kms, amd/drm-kmod/amdgpu@radeonkms
    - e.g. nvidia/nvidia-driver/nvidia@nvidia-modeset
    - e.g. vesa/xf86-video-vesa/vesa, scfb/xf86-video-scfb/scfb
- check graphics hardware

```
# pciconf -lv | grep -B4 VGA
```

- e.g. if it is intel, get drm-kmod

```
# pkg install drm-kmod
```

- test loading the module

```
# kldload i915kms
```

- if no issues, set to load on boot

```
# sysrc kld_list+=i915kms
```

- **note20250107** on 14.2-RELEASE, this has problems
    - install older one, works ok

```
# pkg install drm-515-kmod
```

    - this will remove `drm-kmod`, but it works fine
- some may also need these

```
# pkg install libva-intel-driver mesa-libs mesa-dri
```

dumped...

```
** try this!
```

```
https://wiki.freebsd.org/unitrunker/JWM
```

- enable switching virtual consoles during X session

```
# echo "kern.vty=vt" >> /boot/loader.conf
```

Applications

- development

```
# pkg install -y git geany
```

- console multitasking

```
# pkg install -y screen
```

- office & browser

```
# pkg install -y libreoffice firefox
```

2026/02/09 22:23

Managing FreeBSD

Maintaining the system...

System Upgrade/Update

For example, upgrading 10.1-RELEASE to 10.2-RELEASE

```
# freebsd-update -r 10.2-RELEASE upgrade
```

Then run

```
# freebsd-update install
```

To update within a release, do a

```
# freebsd-update fetch
```

before running 'install'.

Hint hit 'q' when prompted

Package Upgrade/Update

Install package

```
# pkg install <pkg_name> [...]
```

Note: include '-y' to override prompts

Remove package

```
# pkg delete
```

Update catalogue

```
# pkg update
```

Upgrade software

```
# pkg upgrade
```

List installed packages

```
# pkg info
```

Remove all packages and start over

```
pkg delete --all --force
```

Clean all package cache

```
pkg clean
```

If pkg installation quits due to size mismatch or something,

```
pkg clean
rm -rf /var/cache/pkg/*
pkg update -f
```

Using ports system

Will most probably need these at some point...

To get it,

```
portsnap fetch extract
```

To update,

```
portsnap fetch update
```

To manage, use portmaster

Using portmaster

To build portmaster,

```
cd /usr/ports/ports-mgmt/portmaster/ && make install clean
```

To setup portmaster,

```
# cp /usr/local/ etc/portmaster.rc.sample /usr/local/ etc/portmaster.rc
# ee /usr/local/ etc/portmaster.rc
```

To update all ports

```
portmaster -a
```

To search updates

```
portmaster -L | grep "New version available:"
```

To cleanup

```
portmaster --clean-distfiles{-all}
```

To remove port

```
portmaster -e target_port
```

To rebuild port

```
portmaster -r target_port
```

Dumping ground - from portmaster man page...

```
     Build a port locally but use packages for build dependencies, then
delete
     the build dependencies when finished:
       portmaster --packages-build --delete-build-only fooport-1.23

     Update a system using only packages that are available locally:
       portmaster -PP --local-packagedir=<path> -a

     Update all ports that need    updating:
       portmaster -a
```

```
     Update all ports that need    updating, and delete stale distfiles
after the
     update is done:
       1. portmaster -aD
       2. portmaster --clean-distfiles
```

2026/02/09 22:23

Tweaking FreeBSD

Maybe useful to some...

ZeroConf

- avahi & multicast dns

```
pkg install -y avahi nss_mdns
```

- enable avahi daemon in `rc.conf`:

```
avahi_daemon_enable="YES"
```

- modify the `hosts:` line in `nsswitch.conf`:

```
hosts: files dns mdns
```

User Account

- by default root shell is csh and user is sh - edit .shrc to get prettier prompt

```
  PS1="`whoami`@\H:\w\$ "
```

- (IF using slim) to enable x environment after slim login, create `~/.xinitrc` for each user:

```
  exec /usr/local/bin/mate-session
```

Git

1. 'git log' output does not show colorized output
   - can see the escape sequence
   - so, as user, type

```
git config --global core.pager "ls -r"
```

Network Configuration

Note *DHCP client is dhclient - simply run `dhclient <interface>` when booting to console and using something like USB tethering.*

Configure (wireless) network interface (in case was not done during installation)

- FreeBSD has network interface name - based on the driver name (e.g. sis, re)
  - to find this, use `pciconf -lv`
- for normal wired connection (e.g. re0), add to 'rc.conf'

```
ifconfig_re0="DHCP"
```

- for a wireless connection (e.g. ath0), add in

```
wlan_ath0="wlan0"
ifconfig_wlan0="DHCP"
```

- for a secured wireless connection (e.g. WPA protected), add in

```
wlan_ath0="wlan0"
  ifconfig_wlan0="WPA SYNCDHCP"
```

  also, append '/ etc/wpa_supplicant.conf'

```
  network={
      ssid="the_ssid"
      psk="the_psk"
  }
```

- not sure why but `wlan0` was not auto-created?
  - from official handbook, do `ifconfig wlan0 create wlandev ath0`
  - to start up and scan, do `ifconfig wlan0 up scan`
  - or just scan, do `ifconfig wlan0 list scan`

Graphics

Driver for ASUS E5450 Graphics Card (based on Radeon 5450?)

- install driver

```
pkg install -y xf86-video-ati
```

- to load on startup, edit `rc.conf`:

```
kld_list="radeonkms"
```

For Intel Graphics (Asus H81M-K Motherboard)

- install driver

```
pkg install -y xf86-video-intel
```

- to load on startup, edit `rc.conf`:

```
kld_list="i915kms"
```

- install something? (for kernel?)

```
pkg install -y drm-kmod
```

  - some older ones require `drm-fake-kmod` instead

Web Server

Web Server (Apache)

- find apache package(s)

```
pkg search apache2 | grep -e "^apache2"
```

- install apache package(s)

```
pkg install -y php56 mod_php56 php56-mbstring php56-mcrypt php56-zlib
php56-curl php56-gd php56-json
```

- to load on startup, edit `rc.conf`:

```
apache24_enable="YES"
```

- default document path is `/usr/local/www/apache24/data/`

Server Script (PHP)

- find php package(s)

```
pkg search php5 | grep -e "^php5"
```

- install php package(s)

```
pkg install -y apache24
```

- configure `/usr/local/ etc/apache24/Includes/php.conf`:

```
  <IfModule dir_module>
    DirectoryIndex index.php index.html
    <FilesMatch "\.php$">
      SetHandler application/x-httpd-php
    </FilesMatch>
    <FilesMatch "\.phps$">
      SetHandler application/x-httpd-php-source
```

```
    </FilesMatch>
  </IfModule>
```

- copy template configuration file *# cp /usr/local/ etc/php .ini-production /usr/local/ etc/php .ini*

Data-base (mysql/mariadb)

- install package(s)

```
pkg install mariadb103-server mariadb103-client php56-mysqli
```

- to load on startup, edit `rc.conf`:

```
mysql_enable="YES"
```

- by default, it listens to remote request at port 3306. to check:

```
# lsof -i4 -i6
# sockstat -4 -6
```

- to allow local access only, edit `rc.conf`:

```
mysql_args="--bind-address=127.0.0.1"
```

- now, to make sure all is okay:

```
# lsof -i4 -i6 | grep mysql
# netstat -an | grep 3306
# sockstat -4 -6 | grep 3306
```

Label for Partitions/Slices

This is nice to have in / etc/ fstab when device assignment may change (e.g. usb drive on different machine may be assigned differently)

For ufs,

```
# tunefs -L <label> /dev/da0p?
```

To check if assigned,

```
# ls /dev/ufs
```

For swap,

```
# glabel label <label> /dev/da0p?
```

To check if assigned,

```
# ls /dev/label
```

Then, / etc/ fstab entry can be like,

```
/dev/label/<label>    none    swap    sw    0    0
/dev/ufs/<label>  /        ufs        rw    1    1
```

Disk Utility ''gpart''

Show partition

```
# gpart show
```

Resize partition

```
# gpart resize -i 3 da0
```

Not really gpart stuff, but don't forget to grow FS to fit new size

```
# growfs /dev/da0p3
```

Access to Linux ExtFS

At the moment, full R/W access for Ext2, Journal-less for Ext3 and R/O for Ext4.

```
# kldload ext2fs
# mount -t ext2fs /dev/<slice> <mount-path>
```

Creating Disk Layout for Bootable USB

Using GPT

freebsd_gpt_disk_layout.txt

```
creating disk layout (gpt) for freebsd
- assuming disk is da0 (change accordingly for other designation)

0- clean existing partitions/slices

# gpart destroy -F da0

(manual op)
find total sector count using diskinfo
# diskinfo -v da0
```

```
backup/secondary gpt table is at the last 34 sector
# echo '<total> - 34' | bc
<offset>
# dd if=/dev/zero of=/dev/da0 bs=512 seek=<offset>

1- create gpt and bios boot scheme

# gpart create -s gpt da0
da0 created
# gpart add -t freebsd-boot -l gpboot -b 40 -s 1004K da0
da0p1 added
# gpart bootcode -b /boot/pmbr -p /boot/gptboot -i 1 da0
partcode written to da0p1
bootcode written to da0

2- create efi partition

# gpart add -t efi -l gpefiboot -s 127M da0
da0p2 added
# newfs_msdos /dev/da0p2
...

3- copy efi binary

# mount -t msdosfs /dev/da0p2 /mnt
# mkdir -p /mnt/EFI/BOOT
# cp /boot/boot1.efi /mnt/EFI/BOOT/
# umount /mnt

4- create partition/slice for root and swap

# gpart add -t freebsd-ufs -l gprootfs -s 14G da0
da0p3 added
# gpart add -t freebsd-swap -l gpswap da0
da0p4 added

5- format/prepare fs for root

# newfs -U /dev/da0p3
```

Using MBR (just in case... :p)

freebsd_mbr_disk_layout.txt

```
# gpart create -s mbr da0
# gpart bootcode -b /boot/mbr da0
# gpart add -t freebsd da0
# gpart set -a active -i 1 da0
# gpart create -s bsd da0s1
```

```
# gpart bootcode -b /boot/boot da0s1
```

2026/02/09 22:23
2026/02/09 22:23

Extra FreeBSD

Something not-so-common...

Offline Installation

***Note**: Need to test this...* 😛

- Get the desired packages on a FreeBSD machine with internet connection

```
# mkdir /root/offline
# pkg fetch -d -o /root/offline xorg mate slim firefox
```

  - `-d` for dependencies
  - `-o` specifies destination path for the fetched packages
- Copy those files to a portable USB drive
- On the target machine, copy everything to /var/cache/pkg
- Then, do a

```
# pkg -U install xorg mate slim firefox git geany networkmgr
```

  - `-U` is the short form for `–no-repo-update`

USB Thumb-Drive Installation

I want to try to install to a USB thumb-drive... from my FreeBSD virtual machine (VirtualBox). I have a 16GB USB3 Kingston Data Traveller drive, and already installed FreeBSD 12 on a virtual machine.

To prepare the drive layout, checkout here. I'm going to prepare for UEFI boot on a GPT formatted disk.

- plug in usb drive

- find device name (i.e. /dev/???)
- usually da0 is the first usb drive?

- refer to this...

to be continued...

Dual-Boot on UEFI Systems

Find `boot1.efi` as place it in EFI System Partition. It will look for first partition with type freebsd-ufs (which can even be on another disk) and load `loader.efi`.

Install on SSD

Note: *Generally, it seems that this is no longer an issue - some just did a normal install and have no problems at all. But, I want to put this here anyways.*

Creating partitions (from: https://www.wonkity.com/~wblock/docs/html/ssd.html)

```
# gpart create -s gpt ada0
# gpart add -t freebsd-boot -s 1m -a 4k -l ssdboot ada0
# gpart bootcode -b /boot/pmbr -p /boot/gptboot -i1 ada0

# gpart add -t freebsd-ufs -l ssdroot -b 1m -s 4g ada0
# gpart add -t freebsd-ufs -l ssdvarfs -a 1m -s 2g ada0
# gpart add -t freebsd-ufs -l ssdusrfs -a 1m ada0

# newfs -U -t /dev/gpt/ssdrootfs
# newfs -U -t /dev/gpt/ssdvarfs
# newfs -U -t /dev/gpt/ssdusrfs
```

create fstab (save as /tmp/bsdinstall_etc/ fstab)

```
# Device    Mountpoint    FStype    Options    Dump    Pass#
/dev/gpt/ssdroot    /    ufs    rw    1    1
/dev/gpt/ssdvarfs   /var    ufs    rw    2    2
/dev/gpt/ssdusrfs   /usr    ufs    rw    2    2
tmpfs   /tmp    tmpfs    rw,mode=01777    0    0
```

FreeBSD on RasPi400

- got FreeBSD13 arm64 aarch64 image
- write to microsd card
- boot issue
  - ok if boot from usb (note: starting pi4, usb boot is possible!)
  - need to update u-boot binary (look for that in forum)
- console display is not ok when using on tv (high res?)
  - edit config.txt and comment out hdmi_safe=1

going for dwm

- pkg install libX11 libXft libXinerama
- pkg install git

work in progress...

2026/02/09 22:23

From:
https://azman.my1matrix.cc/ - **Azman @MY1**

Permanent link:
https://azman.my1matrix.cc/doku.php?id=freebsd:freebsd

Last update: **2026/02/08 03:46**