

Upgrading FreeBSD

Date Last Edited 01/29/2006

Concept

This document started as a follow up to [The Ultimate Multimedia Server Guide](#) and how to go about keeping your server up to date and patched with the latest O/S patches and security patches. The other reason for this document was to try and create an easy to follow update guide for the not so Unix savvy users that visit my website from time to time. My first time trying to upgrade FreeBSD from sources went well but trying to understand and piece together all the other documentation was more of a daunting task than actually upgrading.

General Information

This guide will lead you through bringing your FreeBSD server up to date to ensure that you get the most entertainment pleasure from your server. Before embarking on this journey I would highly recommend reading this documentation in it's fullest before executing any of it. This guide will require time, patience and attention to detail. This is more time consuming than it is difficult.

You will also notice Blue Font type and Underlined type used throughout this documentation. These are actual www links provided to make finding things a bit easier for you.

Requirements

- 1) Base install of FreeBSD and Ports, preferably FreeBSD 6 as it is the latest stable production release at the time of this document.
- 2) Some sort of Internet connectivity either broadband or some sort of business connectivity like a t1. The faster your downstream connectivity of course the shorter length of time it will take for you to download the sources.
- 3) Your favorite text editor. I prefer VI but there are others like Emacs and Pico.
- 4) sudo or root access to your server and also console access.
- 5) And last but not least a little time and patience.

Installation

The tools that are required for this project can be easily installed from our ports tree. Most tools needed are already installed but there are a couple that we will use to make things a little easier on us and also to automate this process for future use. The main choice of tool for this process is CVS, if you are interested in what is really happening take 10 minutes of your spare time to read [The FreeBSD Handbook CVS section](#).

The first thing we will do is install cvsup-without-gui, you may want to install just cvsup if you have installed X11 on your server. I prefer not to have the overhead of X11 or any type of gui running on my servers.

```
#cd /usr/ports/net/cvsup-without-gui
```

```
# make install clean
```

The above will install cvsup-without-gui and clean the source files out of your ports tree to preserve disk space. CVS will also prompt you asking if it should run MAKE SURE to answer NO to this at this time. There is a tad bit more work to do before running this.

Next we will install a tool that with a little script of my own will help make choosing the fastest cvs server on the Internet for us so that downloading the sources take as little time as possible.

```
#cd /usr/ports/sysutils/fastest_cvsup
# make install clean
```

Now that we have that out of the way we need to prepare our cvsup-file for use. I prefer to have my files in /root for easy backup purposes but you could put them any place you like.

```
# cd ~
# cp /usr/share/examples/cvsup/cvs-supfile /root/cvs-supfile
```

The file you have just copied to your root directory will need to be edited. This file is a very long file so I will not paste it in its entirety here. I would suggest you read your cvs-supfile completely but this document is about making it a tad bit easier so the lines below is what needs editing. Using your favorite editor change these lines to look like such

```
default
*default host=CHANGE_THIS.FreeBSD.org
change to this
*default host=cvsup15.freebsd.org
default
*default prefix=/home/ncvs
change to this
*default prefix=/usr
```

This is the first spot in the documentation where we really make sure this is exact. Scroll all the way down to the bottom of this file and comment out EACH of these lines when you are done it should look like this below

```
## Website
#
# This collection retrieves the www tree of the FreeBSD
# repository
#www
## CVSROOT control files
#
# This is to get the control files that cvs(1) needs and the commit logs.
#cvsroot-all
```

If you are sure everything is correct save the file and exit your editor. At this point you should have a cvs-supfile capable of pulling down the sources, ports tree and documentation for FreeBSD which is

what you need to update your system. Before proceeding we are going to use a little creativity with the great tools provided to us by FreeBSD and automate this process. Now that you have read your cvs-supfile entirely as I suggested you can create your own shorter file thats easier to work with which I **HIGHLY** recommend.

First lets backup the original file so if anything goes wrong you have this to refer back to.

```
# mv /root/cvs-supfile /root/cvs.original
```

Breaking out your favorite editor again create a new shorter file to look like what I have below and call it cvs-supfile.

```
*default host=cvsup15.FreeBSD.org
*default base=/var/db
*default prefix=/usr
*default release=cvs tag=RELENG_6_0
*default delete use-rel-suffix

src-all
*default tag=.
ports-all
doc-all
```

I know this seems like a long winding road but there is day light at the end of this tunnel. You will be so happy knowing you have applied all security and O/S patches to your system that you have made available to all the mischievous activity on the Internet Below you will find my script that I have created to do what we need in one step. By no means do I consider myself a coder, though this script is safe to use and I use it all the time; now if you know of a better way to accomplish this task by all means [share it with](#) the rest of us. I will include it in this document and give you full credit for your work.

In your editor create this file /root/fastestupgrade.sh

```
#!/bin/sh
if SERVER=`fastest_cvsup -q -c us`; then
cvsup -h $SERVER /root/cvs-supfile
# Now let's update the ports database...
/usr/local/sbin/portsdb -Uu
# Now let's see how badly we're out of date...
/usr/local/sbin/portversion -vl "<"
fi
```

Save the file and exit out of your editor. A little explanation; the above script calls the fastest_cvsup executable and searches for the fastest cvsup servers from your location. It then uses that server to run your cvs-supfile to retrieve all of the goodies. Next the script calls portsdb which manages the ports database via the INDEX file. Lastly we call portversion with the -vl switch for verbosity this will list all packages that you have installed and tell you if they are out of date which looks similar to what you see below.

```
samba-3.0.20,1      < needs updating (port has 3.0.21a,1)
```

As you can see at the time I did my upgrade samba was out of date it listed what version I have installed and what the newest version is that has been submitted to ports. Unfortunately there will be a separate document on how to deal with updating your applications that have been installed via ports but

at least now you have some sort of idea of what apps are out of date.

Ok one last tid bit of info on CVS. For those folks that may be on slow links or running low on disk space or simply want to be considered good net citizens you will want to create yourself a refuse file and put it in your base sup directory. With this file you tell cvs to refuse certain files from the server if not you will pull down everything which is not harmful it just takes up extra disk space. I like to pull down everything that is in reference to the English language this way I am not burdening the cvs server in serving me these additional language files, I do not think I will be reading the translated German documentation for FreeBSD anytime soon so :)

```
# cd /var/db/
```

Use your favorite editor to create the refuse file, done this way with VI

```
# vi refuse
```

Now if you only want the English language make sure your refuse file looks like such

```
doc/de
doc/de_*
doc/es
doc/es_*
doc/fr
doc/fr_*
doc/it
doc/it_*
doc/ja
doc/ja_*
doc/nl
doc/nl_*
doc/ru
doc/ru_*
doc/sr
doc/sr_*
doc/zh
doc/zh_*
ports/chinese
ports/french
ports/german
ports/hebrew
ports/japanese
ports/korean
ports/russian
ports/ukrainian
ports/vietnamese
```

Now save the file and exit out of your editor. This will refuse all languages for documents and ports EXCEPT for the English language. FreeBSD is constantly growing and getting new translations to different languages all the time so I recommend browsing [The FreeBSD Repository](#) to check for any new translations but at the time of this writing the above file would work. I will also do my best to try and keep this document up to date as translations do happen. Because I often will refer back to it from time to time also so it will be in my best interest to maintain it.

It is now time for business. We are ready to start our upgrade process now that all the appropriate files are in place. As root

```
# uname -a
```

Make note of the kernel you are currently running you will compare this to the new kernel at the end just for reference.

Now

```
# cd ~
```

```
# sh fastestupgrade.sh
```

This will take a while but after it is complete lets start our world tour.

```
# cd /usr/src/
```

Next we will build world, make our kernel and install our kernel. But we will record this by keeping a log file of everything going on just in case there are errors and if there are you can post your file to the [FreeBSD-Stable Mailinglist](#) for help. Though there are some very complex things going on it is not all that difficult, the FreeBSD group has mastered this process and all usually goes without a hitch.

```
# script ~/buildworld
```

```
# make buildworld
```

You have now built world and captured the process in the /root/buildworld log file. Now cancel the script by hitting cntrl d and you will get an output like below.

```
Script done, output file is /root/buildworld
```

Remember we will keep these log files as insurance for each process.

```
# script ~/buildkernel
```

```
# make buildkernel
```

```
# cntrl d
```

```
Script done, output file is /root/buildkernel
```

```
# script ~/installkernel
```

```
# make installkernel
```

```
# cntrl d
```

```
Script done, output file is /root/installkernel
```

The above steps are for everyone using a generic kernel which I assume is most everyone reading this document. If you are using your own customer kernel already you have already went through these steps and probably know what it is you are doing. If you have inherited this system running a custom kernel reference page 491 of book [FreeBSD Unleashed](#) and it will have everything you need or you can reference the [FreeBSD Handbook](#) either I say is a must for your professional bookshelf.

If you have followed this document step by step all the above steps should have completed without any errors at all, what I like to say is everything went smooth as Stewie Griffiths bottom :) For those adults not familiar treat yourself to a good laugh with the [Adult Cartoon Family Guy](#).

Lets get back to business, now you need to boot your system into single user mode. All of the steps above could have been done through some sort of virtual connection like ssh or vpn connectivity these next steps you will need access to the console itself.

Reboot your machine

```
# shutdown -r now
```

This will shutdown all processes gracefully and began the reboot process. When your system starts to boot back up and you are given the options make sure you choose option 4 so that you boot into single user mode. Now run

```
# fsck -p
```

```
# mount -u /
```

```
# mount -a -t ufs
```

```
# swapon -a
```

This checks the file systems, remounts / read/write, mounts all the other UFS file systems referenced in /etc/fstab and then turns swapping on.

Note: If your CMOS clock is set to local time and not to GMT (this is true if the output of the date(1) command does not show the correct time and zone), you may also need to run the following command: I would do it just to be safe if you are not sure.

```
# adjkerntz -i  
# cd /usr/src
```

This may seem a little scary for the first time but so far it has been a little easier than what it seems. This next piece we will run mergemaster though it is a great tool there seems to be little documentation for it. Of course there is the man page which I would recommend reading. This tool basically compares the files you are about to install in the next step to the files you already have on your system. When running mergemaster you will get options presented to you for each file that looks like such

```
Use 'd' to delete the temporary .name of file  
Use 'i' to install the temporary .name of file  
Use 'm' to merge the temporary and installed versions or parts of them.  
Use 'v' to view the diff results again  
Default is to leave the temporary file to deal with by hand  
How should I deal with this? [Leave it for later]
```

My rules of thumb on this is:

Files I don't want changed which would be files that I have edited by hand like ntpd.conf for setting up time syncing I just hit enter so that I can go back later and compare and edit the files by hand which will not be all that many. Make note of the files you need to go back and compare.

The 'i' is for files that I haven't customized that won't affect things I want my FreeBSD box to do. Most of my files in mergemaster will get this option. Some examples for me are rc.diskless, pam.conf, and rc.syscons. I didn't need to customize these files and their changes won't affect your server install but will need to be there.

Before we proceed we need to take out an insurance policy. As I have stated mergemaster is a very safe tool to use but there is a lot of room for error considering the fuzzy logic that resides between the chair and the keyboard :) We create our insurance policy simply by doing

```
# cp -Rp /etc /etc.old
```

This copies what should be all of your current config files to a backup directory /etc.old. We are now ready to compare the old and new files.

```
# mergemaster -cv -w 120
```

This will run mergemaster with verbosity along with a screen width of 120 instead of the standard 80. If run with the standard 80 sometimes it looks as if the split screen between the files are all one. Proceed through carefully and slowly examining each file keeping in mind my rule of thumb above.

Now that you are done with mergemaster the scariest part of this upgrade, you are ready to wrap things up.

Now run

```
# cd /usr/src  
# script ~/installworld  
# make installworld
```

After this is done with no errors you are complete except for checking your work and bringing the system back up. At this point reboot the system and watch it come up make sure all your startup scripts run and most time they will.

```
# reboot
```

After the system is up take a deep breathe for that sigh of relief, now make sure you booted on the new kernel by executing from sudo or root

```
# uname -a
```

You should now be on the new kernel compare this to the output you took before going through the upgrade. Now try using tools such as ps and top these should be functioning without error. Now go pat yourself on the back and get your favorite beverage, hold your head high with your chest stuck out and feel proud of what you have just accomplished.

References

As I stated when I first completed this task years ago there was a lot of documentation but it was very piece meal. I had a bunch of separate notes in one document all out of place that I used to reference from time to time. As of getting my website together I took those notes and created this documentation as an easy walk through guide. There does seem to be a tad bit more information that you can find with a simple [Goggle](#) search now. But it does seem to be a bit more complicated and scary for the first time upgrader, again the reason for this document.

http://www.taosecurity.com/keeping_freebsd_up-to-date.html

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/cvsup.html

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/makeworld.html

Must have books for your bookshelf, I am not associated with any of these but it does help support the opensource movement, which gave us the capability for this great server we have created.

<http://www.bsdmall.com/freebun.html>

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/

<http://www.bsdmall.com/cofr.html>

Hints and Tips

It is always recommended that you keep a good backup. I was blessed with a NAS as a gift from my wife that serves this purpose for me. It is connected to my network and I use the utility rsync to backup to this device. It really works out great because very little hours get put on this drive as it is only on when I am doing my weekly backup which is good enough for me for a home server. A usb drive could also serve this function. I would surely recommend that you come up to speed on rsync and start using it. You can get started by visiting

<http://samba.anu.edu.au/rsync/>

And though this gentleman wrote this for an older version of FreeBSD it will work with your install and it is great easy to follow documentation

<http://caia.swin.edu.au/reports/020927A/>

Questions/Comments

[Drop me a line](#)

This document written and maintained by Elijah Savage at <http://www.digitalrage.org> with OpenOffice 2.0 and exported to PDF,Doc, and rtf formats for versatility and ease of use for the end user. OpenOffice what a great app.

Download this Document

[PDF Format](#)

[Word 97/2000/XP Doc Format](#)

[OpenOffice Format](#)

[Rich Text Format](#)