

The Ultimate Media Server

Apache+SSL, PHP, MySQL and Jinzora

Date Last Edited 01/29/06 Rev.1

Concept

I started out on this project to create the ultimate multimedia server using all opensource tools and applications. Some of you may be thinking along the lines of a TIVO device, no this was to create a hosting server on my business broadband connection so that I can have access to and listen to my MP3's from anywhere that had an Internet connection while keeping it all secure as possible.

General Information

This guide will lead you through creating a secure ssl based webserver to be able to stream your multimedia across the World Wide Web. Before embarking on this journey I would highly recommend reading this documentation in it's fullest before executing any of it. You may find some pointers in the tips and tweaks section that you can make during installation that would make this install even easier and make it a one time install.

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.
- 2) A public domain name for WWW access.
- 3) Some sort of Internet connectivity either broadband or some sort of business connectivity like a t1. I would like to point out and recommend that your upstream bandwidth be at least 384kbps.
- 4) Your favorite text editor. I prefer VI but there are others like Emacs and Pico.
- 5) A Very large hard drive or at least large enough to hold all of your media for your server.

Installation

Now lets get to the fun piece of this document. As I stated this guide is based on FreeBSD 6 which you should have installed, there is so much documentation on the installation of FreeBSD that I will not guide you through that piece, not to mention it is one of the easiest UNIX's to install. But if you do find that you need a bit of help you can always visit [The FreeBSD Handbook](#). After base install I did make sure I had the latest version of these applications in ports by using CVS to get the latest and greatest ports collection. This process of [Updating Ports](#) is also a very easy task and well documented by the FreeBSD organization

The first thing we will embark on is installing Apache1.3 and SSL. Apache currently has 3 development paths Apache1.3 and Apache2.0 and the recently released Apache2.2, at the time of this

writing I still preferred to run Apache1.3 call me old school, this version has been around for a very long time and is the most used web server on the Internet today and the Apache staff is still developing security patches for it. You can visit [Apache's Website](#) to see what the differences are in versions.

Install Apache1.3 and SSL from ports

Login to your server via the console or ssh and make sure you have root privileges. All text with black background are actually commands executed on the server.

```
# cd /usr/ports/www/apache13-modssl
# make install
```

The above will install apache13-modssl in /usr/local/etc/apache/

```
# echo 'apache_enable="YES"' >> /etc/rc.conf
# echo 'apache_flags="-DSSL"' >> /etc/rc.conf
```

The above commands makes sure that apache will startup on bootup. See I told you it was not that hard all it takes is a little time and patience and desire to follow through. Now moving on to installing MySQL Sever

Install MySQL Server with SSL Support

```
# cd /usr/ports/databases/mysql41-server
# make install WITH_OPENSSL=yes
```

Go grab a coke or your favorite beverage as this may take a while but when you are done you will have just installed MySQL Server with SSL support successfully.

Next we will make sure that MySQL Server starts up on bootup

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

Next we go about creating the root password for MySQL which should **NOT** be the same password as the system root user. You can do this with one command from the command line which is what I use below. There is also another way of doing this which can be found in the MySQL documentation, I am taking the easy way out since I am in the comfort of my own home with no one looking over my shoulder.

```
# mysqladmin -u root password newpassword
```

We will now create the default database

```
# mysql_install_db -user=mysql
```

Thats it for MySQL, this rocks does it not?

PHP Support

When completing the steps below a php configuration screen will popup don't worry this is correct and you need to make sure that you check the OpenSSL box and GD Image support box.

```
# cd /usr/ports/www/mod_php4
# make install clean
# cd /usr/ports/lang/php4-extensions
# make install clean
```

The above installs php4 support needed by our multimedia streaming server and Apache.

Next we need to edit the apache configuration file to instruct it to Load the php modules. Add the two lines below to the httpd.conf file after the "LoadModule" lines but within the same section to give

apache php support.

```
# vi /usr/local/etc/apache/httpd.conf
AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps
```

This next section is the hardest part of this installation but only because it requires a bit more attention to detail simply due to the number of questions that will be asked during installation. So if you are getting a bit tired I would recommend taking a break here and coming back later. If your ready to or feel like continuing lets go for it because you are just about $\frac{3}{4}$ the way to the end.

Generate Self Signed Certificate

Here we will be creating a self sign certificate to be used by our apache server. You can create a certificate and have it signed by a Public Authority like [Verisign](#) or [Thawte](#), the services provided by these companies do cost unlike self signing your own certificate. There is definitely an advantage to having a publically sign certificate by one of the above mentioned companies and that being Trust and Security. Though for this type of installation I am perfectly comfortable using self signed.

Change directory to the spot where you would like to save your certificate I chose the root directory as this place, but from here I will copy it to the proper place for apache to use it.

```
# cd ~
# openssl genrsa -des3 -out server.key 1024
```

At this point you asked to enter a password. Please make sure that you remember or **TEMPORARILY** write down the password because you will need it again.

```
# openssl req -new -key server.key -out server.csr
```

Here you have created the CSR (Certificate Signing Request) for the key that you created in the previous step. It should have asked you for a password which would be the one you used from the previous step. You will also be asked a number of other questions during this process the most important being “Common Name” make sure that you use your FQDN for this step. Example, my certificate is for <https://www.digitalrage.org> so your “Common Name” should be entered as www.digitalrage.org. If you manage to get ahead of yourself and did not do this properly thats ok because you can go back and run through this step again. Now you are ready to self sign your certificate and validate it for 365 days.

```
# openssl x509 -req -days 365 -in /root/server.csr -signkey /root/server.key -out /root/server.crt
```

Now we need to copy the files to the appropriate directory

```
# cp ~/server.key /usr/local/etc/apache/ssl.key/
# cp ~/server.crt /usr/local/etc/apache/ssl.crt/
```

You made it through to the end of creating and signing your certificate and putting it in place for apache server to be able to read and utilize it. I would like to suggest that on your spare time you actually read through the [documentation and FAQ](#) so that you understand a bit more about certificates and what it is you have just accomplished. You have just installed everything you need to get to the exciting piece we have all been working towards. Which is installing [Jinzora](#) the application that will give us the ultimate web based interface for streaming our favorite music across the internet to that lonely hotel room you occupy when on business travel or that interface for serving your media to any pc in your home.

Installation of Jinzora

You begin by downloading the tar.gz file from [Jinzora's website](#). I downloaded the tar file to my /tmp directory. I then went about untaring it and putting it in the htdocs/root directory of apache which by default is /usr/local/www/data/.

```
# cd /tmp
# tar -zxvf jinzora-X.X.tar.gz
# mkdir /usr/local/www/data/jinzora/
# cp -R /jinzora/ /usr/local/www/data/jinzora/
```

Next we need to make sure the proper permissions are set on the jinzora directory above we created you do this by running the script the jinzora staff has so kindly included to make sure all files and permissions are set properly and securely.

```
#cd /usr/local/www/data/jinzora
# sh configure.sh
```

We are just about there we need to create our jinzora database, start our services then finish up.

Here we need to create our database that Jinzora will use, you will need to remember what you set your MySQL root users password to.

```
# mysql -p
# mysql> create database jinzora;
Query OK, 1 row affected (0.00 sec)
# mysql> grant select,insert,update,create,delete
  -> on jinzora.*
  -> to jinzorauser@'localhost'
  -> identified by 'password';
Query OK, 0 rows affected (0.04 sec)
mysql> \q
```

Next we need to tie it all together. We need to start our services Apache and MySQL Server so that you can finish the install of Jinzora.

```
# rehash
# apachectl startssl
# /usr/local/etc/rc.d/mysql-server start
```

These should start up without any issues but if not more than likely it will be a permissions error just make sure that you pay attention to the error and correct it before moving on but if this is a fresh installation as this guide assumes then there will not be any.

All that there is left to do now is to make sure you have copied all your media to your server which I have put in /media and then point your favorite web browser to <http://yoursite/path/to/jinzora/index.php> in my case that is <https://www.digitalrage.org/jinzora/index.php>. From here on out it is gui based via your browser and very simple. The [Jinzora staff](#) has put in a lot of hard work making this as simple as possible. Just make sure that you follow their instructions and on screen prompts and you should be up and running within 5 minutes. If you do find yourself needing some sort of support there is a [support](#)

[forum](#) and also [premium support](#) at a very reasonable price provided by the jinzora staff.

References

This document was created using a number of different website and applications. If you care to expand your knowledge about the base server or the applications used, here are a few links to get you started.

<http://www.jinzora.com>

<http://www.daemonnews.org/>

<http://www.freebsd.org>

<http://www.freebsdjournal.org/>

<http://httpd.apache.org/>

<http://dev.mysql.com/doc/>

<http://www.php.net/>

Server Configuration Tips/Living Document

This part of the document I like to refer to as the Tweaks section or Living Document section because it will surely be changing as advances or tweaks are made and comments come in from the general public. First and foremost I would surely visit the [Tips and Tweaks Jinzora forum](#) every now and again for the latest and greatest discoveries.

1) By default apache logs to one file and there is this big misconception that it is a very difficult task to rotate the logs at a specific time or given interval to keep one single file from growing enormously and being hard to manage. Every version of apache comes with rotatelog, on this machine you will find it in /usr/local/sbin/rotatelog. You will need to edit your /usr/local/etc/apache/httpd.conf and comment out these lines

```
CustomLog /var/log/httpd-access.log combined
```

```
TransferLog /var/log/httpd-access.log
```

To look like

```
CustomLog "|rotatelog /var/log/apachelog 86400" combined
```

```
TransferLog "|rotatelog /var/log/apachelog 86400"
```

This is basically using an external program bundled with apache and doing what is called piped logging and rotating your log file ever 24 hours. More details can be found in the [Apache Documentation](#).

2) I would recommend that before installation of jinzora that you organize your media/mp3's in the format of /genre/artist/album title/track. Then during installation of jinzora make sure you choose the file system base installation and not the id3 base installation doing it this way will give you speed advantages and better control vice using the id3 format. Not to mention if you are not good about ripping your cd's and making sure the id3 tags are right you will end up with one slow install and a mess of a website.

3) During installation of php at the very end you were told to copy /usr/local/etc/php.ini-dist to /usr/local/etc/php.ini make sure that you do not skip this step. PHP is a very important part to Jinzora as it is written in this language. I have noticed on very large installations of Jinzora meaning a site with what I would classify as large being over 5Gig in media data, that you will need to make changes to

this file. After copying the above file you will need to make sure that you edit it and change the section you see below. Of course these are the parameters I use and are dependent on how much memory you have in your machine so these parameters may not be optimal for you. My machine has 1gig of memory and this has turned out to be the sweet spot for me and Jinzora simply screams.

```
.....
```

```
; Resource Limits ;
```

```
.....
```

```
max_execution_time = 300 ; Maximum execution time of each script, in seconds
```

```
max_input_time = 60 ; Maximum amount of time each script may spend parsing request data
```

```
memory_limit = 64M ; Maximum amount of memory a script may consume (8MB)
```

4) By default MySQL will run great but there are some tweaks that I have found to be very helpful especially in reference to speeding up jinzora. MySQL has included four template files that you can use for tweaking, they are

my-huge.cnf, my-large.cnf, my-medium.cnf, my-small.cnf these file reference the type of system you have just installed MySQL on and the type of utilization you expect the database to handle. I copied the /usr/local/share/mysql/my-medium.cnf file to /var/db/mysql/my.cnf which is where MySQL will expect to find this file for use. I then made the following tweaks to this file and MySQL which can be seen below. But before tweaking this file I recommend that you at least read the [Getting Started MySQL documentation](#) because if you are not sure what it is you are doing you can bring your system to it's knees with the wrong values. The [AdminZone](#) has a very popular thread about tweaking MySQL for speed also but is starting to age a bit. But thats ok because a quick google search for this topic would give you plenty of information.

Configuration changes I made to my my.cnf file that really made a speed difference in Jinzora

```
key_buffer = 64M
```

```
max_allowed_packet = 16M
```

```
table_cache = 1024
```

```
join_buffer_size = 2M
```

```
sort_buffer_size = 4M
```

```
read_buffer_size = 2M
```

```
myisam_sort_buffer_size = 64M
```

```
thread_cache = 8
```

```
# Try number of CPU's*2 for thread_concurrency
```

```
thread_concurrency = 2
```

```
max_connections = 200
```

```
max_user_connections = 200
```

```
query_cache_limit = 1M
```

```
query_cache_size = 32M
```

```
query_cache_type = 1
```

5) Make a few apache tweaks for low utilized sites so it does not eat up so many resources, though the default is good I prefer to change a couple of settings.

```
# vi /usr/local/etc/apache/httpd.conf
# KeepAliveTimeout 5
# MaxClients 75
# ServerAdmin webmaster@yourdomain.com
# ServerName www.yourdomain.com
```

This setting is Default but make sure it is set this way so that apache does not use resources or get bogged down trying to host lookups for everyone that visits your site.

```
# HostnameLookups Off
```

Also when starting apache with SSL you will be prompted to enter the password you used when creating the key. The reason you have to do this every time is because the key is stored in an encrypted format. *****One BIG thing to note here is that upon reboot the box will NOT boot up all the way until you put in the pass phrase.***** So if this box is hosted remotely and you have no console to it I would recommend getting console access to it via some console server. If this is not possible then as a last resort you can remove the encryption off the key. But please know that this is not recommended as anyone with access to the server if it is not secure can get your key and impersonate you. If you are sure you wish to do this then follow the steps below.

```
# cd /usr/local/etc/apache/ssl.key
# cp server.key server.key.orig
# openssl rsa -in server.key.orig -out server.key
```

6) During this installation all these packages were installed from the ports packages. I had tried to include in this documentation the process for upgrading all ports packages with CVS before installing but it turned out way to long. The FreeBSD Handbook has a very easy walk through of installing cvs and for upgrading ports. But I have custom wrote some a few simple scripts that makes this process a little easier which will be in a different document to come. So stay tuned and keep check for the [FreeBSD Upgrading Ports](http://www.digitalrage.org) document at <http://www.digitalrage.org>

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](#)

Revisions

Rev1. [Upgrading FreeBSD Doc](#) created and linked to from this document.