FreeBSD Administration - Support #433

Setup a FreeBSD, Nginx, MariaDB 5.5, PHP5 Web Server

08/09/2014 06:51 PM - Daniel Curtis

Status:	Closed	**Start date:**	05/02/2014
Priority:	Normal	**Due date:**	
Assignee:	Daniel Curtis	**% Done:**	100%
Category:	Web Server	**Estimated time:**	3.00 hours
Target version:	FreeBSD 9	**Spent time:**	7.50 hours

Description

Here is a procedure to install a FreeBSD with Nginx, MariaDB and PHP server stack. If any version of the packages needs to be changed, replace the versions in the commands accordingly.

Pre-installation requirements

- Before installation of the components, make sure everything is up to date using the following command:

```
pkg update -f && pkg upgrade
```

- Install portmaster:

```
cd /usr/ports/ports-mgmt/portmaster
make install clean
pkg2ng
```

- Edit the /etc/hosts file

```
vi /etc/hosts
```

  - And add/modify the following line:

```
192.168.1.100                www.example.com
```

Install Nginx

- Install Nginx

```
portmaster www/nginx
```

- Start and enable nginx at boot:

```
echo 'nginx_enable="YES"' >> /etc/rc.conf
service nginx start
```

- Create a configuration directory to make managing individual server blocks easier

```
mkdir /usr/local/etc/nginx/conf.d
```

- Edit the main nginx config file:

```
vi /usr/local/etc/nginx/nginx.conf
```

  - And strip down the config file and add the include statement at the end to make it easier to handle various server blocks:

```
#user  nobody;
worker_processes  1;
error_log  /var/log/nginx-error.log;

events {
  worker_connections  1024;
}

http {
  include       mime.types;
  default_type  application/octet-stream;

  sendfile        on;
  #tcp_nopush     on;

  #keepalive_timeout  0;
  keepalive_timeout  65;

  #gzip  on;

  # Load config files from the /etc/nginx/conf.d directory
  include /usr/local/etc/nginx/conf.d/*.conf;

}
```

Create a Default Static Website

Start by setting up a simple static website, no server-side stuff PHP or Ruby; just plain HTML, CSS, JavaScript, etc.

- Create a directory for the web site:

```
mkdir /usr/local/www/www.example.com
```

- Add a **default site server block**:

```
vi /usr/local/etc/nginx/conf.d/default.conf
```

  - Add the following:

```
  server {
    listen       80 default_server;
    server_name  www.example.com;

    access_log  /var/log/www.example.com.log  main;

    location / {
      root   /usr/local/www/www.example.com;
      index  index.html index.htm;
    }

    # redirect server error pages to the static page /50x.html
    error_page   500 502 503 504  /50x.html;
```

```
    location = /50x.html {
      root   /usr/local/www/nginx-dist;
    }

  }
```

Install PHP

The PHP support in FreeBSD is extremely modular so the base install is very limited. It is very easy to add support using the *lang/php5-extensions* port. This port provides a menu driven interface to PHP extension installation. Alternatively, individual extensions can be installed using the appropriate port.

- Install PHP5 and other supporting packages:

```
portmaster lang/php5
```

- Install PHP extensions and a few modules:

```
portmaster lang/php5-extensions databases/php5-mysql databases/php5-mysqli databases/php5-pdo_
mysql www/php5-session
```

NOTE: There are many more PHP modules, to search for more PHP modules run:

```
find /usr/ports/ -name "php5-*"
```

NOTE: PHP capabilities can be further extended by using PECL packages, to search for more PECL packages run:

```
find /usr/ports/ -name "pecl-*"
```

- Configure the default PHP settings

```
cp /usr/local/etc/php.ini-production /usr/local/etc/php.ini
```

Configure PHP-FPM

- Edit /usr/local/etc/php-fpm.conf:

```
vi /usr/local/etc/php-fpm.conf
```

  - Make the following changes:

```
events.mechanism = kqueue
listen = /var/run/php-fpm.sock
listen.owner = www
listen.group = www
listen.mode = 0666
```

- Start and enable PHP-FPM at boot:

```
echo 'php_fpm_enable="YES"' >> /etc/rc.conf
service php-fpm start
```

- Restart nginx:

```
service nginx restart
```

Create a PHP Website

- Create a directory for the web application:

```
mkdir /usr/local/www/phpapp.example.com
```

- Add a **phpapp.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/phpapp.example.com.conf
```

  - Add the following:

```
server {
  listen       80;
  server_name  phpapp.example.com;
  root         /usr/local/www/phpapp.example.com;
  access_log   /var/log/phpapp.example.com-access.log;
  error_log    /var/log/phpapp.example.com-error.log

  location / {
    index  index.php index.html index.htm;
  }

  # For all PHP requests, pass them on to PHP-FPM via FastCGI
  location ~ \.php$ {
    fastcgi_pass unix:/var/run/php-fpm.sock;
    fastcgi_param SCRIPT_FILENAME /usr/local/www/phpapp.example.com$fastcgi_script_name;
    fastcgi_param PATH_INFO $fastcgi_script_name;
    include fastcgi_params; # include extra FCGI params
  }

}
```

Install Phusion Passenger

- Reinstall Nginx with Passenger support

```
portmaster www/rubygem-passenger
```

NOTE: Make sure to enable **[X]NGINX** when running make config-recursive on *rubygem-passenger*
NOTE: Make sure to enable **[X]PASSENGER** when running make config on *nginx*
NOTE: Ruby capabilities can be further extended by using rubygem packages, to search for more packages run:

```
find /usr/ports/ -name "rubygem-*"
```

Configure Passenger

- Edit the main nginx config file:

```
vi /usr/local/etc/nginx/nginx.conf
```

- And add the Passenger config parameters:

```
#user  nobody;
worker_processes  1;
error_log  /var/log/nginx-error.log;

events {
  worker_connections  1024;
}

http {
  include       mime.types;
  default_type  application/octet-stream;

  sendfile        on;
  #tcp_nopush     on;

  #keepalive_timeout  0;
  keepalive_timeout  65;

  #gzip  on;

  # Load Phusion Passenger module globally
  passenger_root /usr/local/lib/ruby/gems/2.1/gems/passenger-5.0.6;
  passenger_ruby /usr/local/bin/ruby21;
  passenger_max_pool_size 15;
  passenger_pool_idle_time 300;

  # Load config files from the /etc/nginx/conf.d directory
  include /usr/local/etc/nginx/conf.d/*.conf;

}
```

Create a Ruby Website

- Create a directory for the web application:

```
mkdir /usr/local/www/rubyapp.example.com
```

- Add a **rubyapp.example.com server block**:

```
vi /usr/local/etc/nginx/conf.d/rubyapp.example.com.conf
```

- Add the following:

```
server {
  listen       80;
  server_name  rubyapp.example.com;
  root         /usr/local/www/rubyapp.example.com/public;
  access_log   /var/log/rubyapp.example.com-access.log;
  error_log    /var/log/rubyapp.example.com-error.log

  passenger_enabled on;
  passenger_user    www;
  passenger_group   www;

}
```

Securing Nginx With SSL

- Install OpenSSL:

```
portmaster security/openssl
```

Enabling SSL in Nginx is simple. First add the ssl directive in the server listen option, then add the SSL certificate and key paths.

- The basic SSL server block should be look similar to the following:

```
    server {
        listen              443 ssl;
        server_name         www.example.com;
        ssl_certificate     www.example.com.crt;
        ssl_certificate_key www.example.com.key;
        ...
    }
```

- Setup the Diffie-Hellman Key Exchange Parameters

```
cd /usr/local/etc/nginx
openssl dhparam -out dhparam.pem 4096
```

- Generate a strong SSL key and a CSR to send for signing by a CA:

```
cd /usr/local/etc/nginx
openssl req -sha512 -out www.example.com.csr -new -newkey rsa:4096 -nodes -keyout www.example.com.key
```

    - If the received SSL certificate requires additional bundle certificates, add them together like so:

```
cd /usr/local/etc/nginx
cat www.example.com.crt www.example.com.bundle > www.example.com.chained.crt
```

- Setup the default site configuration:

```
vi /usr/local/etc/nginx/conf.d/www.example.com.conf
```

    - Then add or modify the configuration to look similar to the following:

```
server {
  listen 80;
  listen 443 default ssl;
  server_name www.example.com;

  # Turn on ans set SSL key/cert
  ssl on;
  ssl_certificate /usr/local/etc/nginx/www.example.com.crt;
  ssl_certificate_key /usr/local/etc/nginx/www.example.com.key;

  # Strong SSL configuration
  ssl_ciphers 'AES128+EECDH:AES128+EDH:!aNULL';
  ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
```

```
    ssl_session_cache  builtin:1000  shared:SSL:10m;
    ssl_stapling on;
    ssl_stapling_verify on;
    ssl_prefer_server_ciphers on;
    ssl_dhparam /usr/local/etc/nginx/dhparam.pem;
    add_header Strict-Transport-Security max-age=63072000;
    add_header X-Frame-Options DENY;
    add_header X-Content-Type-Options nosniff;

    root /usr/local/www/;
    index index.html index.htm;
    autoindex on;

    # Uncomment to force HTTPS
#   if ($scheme = http) {
#     return 301 https://$server_name$request_uri;
#   }

}
```

Certificate Bundles

Some browsers may complain about a certificate signed by a well-known certificate authority, while other browsers may accept the certificate without issues. This occurs because the issuing authority has signed the server certificate using an intermediate certificate that is not present in the certificate base of well-known trusted certificate authorities which is distributed with a particular browser. In this case the authority provides a bundle of chained certificates which should be concatenated to the signed server certificate.

- The server certificate must appear before the chained certificates in the combined file:

```
cat www.example.com.crt bundle.crt > www.example.com.chained.crt
```

- The resulting file should be used in the ssl_certificate directive:

```
    server {
        listen              443 ssl;
        server_name         www.example.com;
        ssl_certificate     www.example.com.chained.crt;
        ssl_certificate_key www.example.com.key;
        ...
    }
```

Install MariaDB

- Install MariaDB 5.5 server and client

```
portmaster databases/mariadb55-server databases/mariadb55-client
```

Configure MariaDB server

- Configure the MariaDB server

```
cp /usr/local/share/mysql/my-small.cnf /usr/local/etc/my.cnf
```

  - **my-small.cnf** - for systems with up to 64 Mb of RAM.
  - **my-medium.cnf** - for systems with up to 128 Mb of RAM (ideal for web servers).
  - **my-large.cnf** - for systems with 512 Mb of RAM (dedicated MySQL servers).

- **my-huge.cnf** - for systems with 1-2 Gb of RAM (datacentres etc.).

- Enable MariaDB to start at boot:

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

- Start MariaDB

```
service mysql-server start
```

- Set password for mysql using the following command

```
mysqladmin -uroot password
```

- Restart mysql using the following commands:

```
service mysql-server restart
```

Install and configure phpMyAdmin

- Install phpmyadmin:

```
pkg install phpmyadmin
```

- Setup phpMyAdmin for nginx by adding the following to the server{ } block in /usr/local/etc/nginx/nginx.conf:

```
## phpMyAdmin
location ^~ /phpmyadmin {
  access_log  off;
  rewrite ^  /phpMyAdmin/ permanent;
}

location /phpMyAdmin {
  root /usr/local/www/phpMyAdmin;
  index index.php index.html;

  ## Only Allow connections from localhost
  allow 127.0.0.1;
  deny all;

    location ~ ^/phpMyAdmin/(.*\.php)$ {
      root /usr/local/www/phpMyAdmin;
      fastcgi_pass unix:/var/run/php-fpm.sock;
      fastcgi_param SCRIPT_FILENAME /usr/local/www/phpMyAdmin$fastcgi_script_name;
      fastcgi_param PATH_INFO $fastcgi_script_name;
      include fastcgi_params; # include extra FCGI params
    }
}
```

Now its time to configure phpMyAdmin. Do this by creating the file /usr/local/www/phpMyAdmin/config.inc.php, the basic configuration file for phpMyAdmin. Traditionally, users have manually created or modified /usr/local/www/phpMyAdmin/config.inc.php, but now phpMyAdmin includes a nice setup script, making it much easier to create this file with the settings you want.

- Start by creating the directory /usr/local/www/phpMyAdmin/config and make it writable by the phpMyAdmin setup script:

```
mkdir /usr/local/www/phpMyAdmin/config
chmod o+w /usr/local/www/phpMyAdmin/config
```

- Then make /usr/local/www/phpMyAdmin/config.inc.php readable by the phpMyAdmin setup script:

```
chmod o+r /usr/local/www/phpMyAdmin/config.inc.php
```

- Now open your web browser and navigate to http://www.example.com/phpmyadmin/setup where you will see the phpMyAdmin setup *Overview* page.
- Select **New Server** and then select the **Authentication** tab.
  1. Under the **Authentication type** choose http from the drop-down list (using HTTP-Auth to sign-in into phpMyAdmin will avoid storing login/password credentials directly in config.inc.php)
  2. And remove root from the **User for config auth**.
- Now select **Apply** and you will be returned you to the Overview page where you should see a new server listed.
- Select **Save** again in the Overview page to save your configuration as /usr/local/www/phpMyAdmin/config/config.inc.php.
- Now let's move that file up one directory to /usr/local/www/phpMyAdmin where phpMyAdmin can make use of it.

```
mv /usr/local/www/phpMyAdmin/config/config.inc.php /usr/local/www/phpMyAdmin
```

- Now let's try out phpMyAdmin to make sure it works. Point your web browser to http://www.example.com/phpmyadmin where you will be presented with a pop-up box requesting you to log in.
  Use "root" and the MySQL password you set up previously, then you should be directed to the phpMyAdmin administration page.
- We no longer need the /usr/local/www/phpMyAdmin/config directory so let's remove it, and the read permission we added previously to /usr/local/www/phpMyAdmin/config.inc.php:

```
rm -r /usr/local/www/phpMyAdmin/config
chmod o-r /usr/local/www/phpMyAdmin/config.inc.php
```

- And wrap up by restarting the nginx and MySQL servers:

```
service nginx restart
service mysql-server restart
```

Resources

- http://www.bsdnow.tv/tutorials/nginx
- http://forums.freebsd.org/viewtopic.php?t=30268

Related issues:

Copied from FreeBSD Administration - Support #432: Install A FreeBSD, Apache ...	**Closed**	**05/02/2014**

History

#1 - 08/09/2014 06:51 PM - Daniel Curtis

- Copied from Support #432: Install A FreeBSD, Apache 2.4, MariaDB 5.5, PHP 5 (FAMP) Server added

#2 - 08/15/2014 12:41 PM - Daniel Curtis

- Description updated

- % Done changed from 80 to 90

#3 - 08/17/2014 03:51 PM - Daniel Curtis

- Description updated

- Status changed from New to In Progress

#4 - 08/26/2014 04:29 PM - Daniel Curtis

- Description updated

#5 - 08/26/2014 04:40 PM - Daniel Curtis

- Description updated

#6 - 12/12/2014 09:23 AM - Daniel Curtis

- Description updated

- Status changed from In Progress to Resolved

- % Done changed from 90 to 100

#7 - 12/12/2014 09:32 AM - Daniel Curtis

- Description updated

#8 - 12/12/2014 09:32 AM - Daniel Curtis

- Description updated

#9 - 12/12/2014 09:33 AM - Daniel Curtis

- Description updated

#10 - 12/12/2014 10:02 AM - Daniel Curtis

- Status changed from Resolved to Closed

#11 - 02/09/2015 02:44 PM - Daniel Curtis

- Description updated

#12 - 02/11/2015 01:58 PM - Daniel Curtis

- Description updated

#13 - 02/14/2015 10:37 AM - Daniel Curtis

- Target version set to FreeBSD 9

#14 - 02/14/2015 11:44 AM - Daniel Curtis

- Subject changed from Setting Up A FreeBSD, Nginx, MariaDB 5.5, PHP 5 Web Server Stack to Setting Up A FreeBSD, Nginx, MariaDB 5.5, PHP5 Server

- Category set to Web Server

#15 - 04/14/2015 04:48 PM - Daniel Curtis

- Description updated

#16 - 04/16/2015 05:06 PM - Daniel Curtis

- Description updated

- Status changed from Closed to In Progress

#17 - 04/18/2015 04:57 AM - Daniel Curtis

- Subject changed from Setting Up A FreeBSD, Nginx, MariaDB 5.5, PHP5 Server to Setup a FreeBSD, Nginx, MariaDB 5.5, PHP5 Web Server

- Description updated

#18 - 06/06/2015 10:44 AM - Daniel Curtis

- Copied to Support #622: Setup a Nginx, PostgreSQL, PHP 5.6 Web Server on FreeBSD added

#19 - 06/06/2015 10:44 AM - Daniel Curtis

- Copied to deleted (Support #622: Setup a Nginx, PostgreSQL, PHP 5.6 Web Server on FreeBSD)

#20 - 06/09/2015 04:52 PM - Daniel Curtis

- Description updated

- Status changed from In Progress to Resolved

#21 - 06/09/2015 04:59 PM - Daniel Curtis

- Description updated

#22 - 06/09/2015 05:00 PM - Daniel Curtis

- Description updated

#23 - 06/09/2015 05:02 PM - Daniel Curtis

- Description updated

#24 - 06/09/2015 05:03 PM - Daniel Curtis

- Description updated

#25 - 06/11/2015 09:54 AM - Daniel Curtis

- Description updated

#26 - 06/12/2015 10:31 AM - Daniel Curtis

- Status changed from Resolved to Closed