How to Install Apache, MySQL, PHP (FAMP Stack) on FreeBSD 14.0

Introduction

FreeBSD, Apache, MySQL, and PHP (FAMP stack) is a collection of open-source software applications that manage the runtime and development of dynamic web applications on a server. Apache works as a web server to delivers web application files while MySQL functions as the database backend, and PHP processes dynamic web application contents.

This article explains how to install the Apache, MySQL, and PHP (FAMP) stack on FreeBSD 14.0 and deliver dynamic web applications on your server.

Prerequisites

Before you begin:

- Deploy a FreeBSD 14.0 instance on Vultr.
- Create a new domain **A** record pointing to the instance IP address. For example, `app.example.com`.
- Access the server using SSH.
- Create a non-root user with sudo privileges and switch to the user.
- Update the server.

Install Apache

Apache is available in the default repositories on FreeBSD 14.0 with the latest package information. Follow the steps below to install the latest Apache web server package and enable the application to run on your server.

1. Update the server packages catalog.

   console Copy

   ```
   $ sudo pkg update
   ```

   Explain Code

2. Install Apache on your server.

   console Copy

   ```
   $ sudo pkg install -y apache24
   ```

   Explain Code

3. Verify that Apache is installed on your server.

   console Copy

   ```
   $ apachectl -v
   ```

   Explain Code

   Output:

   ```
   Server version: Apache/2.4.59 (FreeBSD)
   Server built:   unknown
   ```

4. Enable Apache to automatically start at boot.

   console Copy

   ```
   $ sudo service apache24 enable
   ```

   Explain Code

5. Start the Apache web server.

   console Copy

   ```
   $ sudo service apache24 start
   ```

   Explain Code

6. View the Apache service status and verify that the web server is running.

   console Copy

   ```
   $ sudo service apache24 status
   ```

Explain Code

Output:

```
apache24 is running as pid 2536.
```

7. Access your server IP address using a web browser such as Chrome and verify that the default Apache virtual host web page displays.

```
http://SERVER-IP
```

Default Apache webpage

Image not found or type unknown

Install MySQL

MySQL is available in the default FreeBSD 14.0 repositories with multiple versions. Follow the steps below to install the latest MySQL database server version and enable it to run on your server.

1. Search all MySQL versions available in the default FreeBSD repositories.

   console Copy

```
$ sudo pkg search mysql
```

   Explain Code

   Output:

```
...
mysql80-client-8.0.35          Multithreaded SQL database (client)
mysql80-server-8.0.35_1        Multithreaded SQL database (server)
mysql81-client-8.1.0           Multithreaded SQL database (client)
mysql81-server-8.1.0           Multithreaded SQL database (server)
...
```

2. Install the latest MySQL server and client packages.

   console Copy

```
$ sudo pkg install mysql81-server mysql81-client
```

   Explain Code

3. View the installed MySQL version on your server.

   console Copy

```
$ mysql --version
```

Explain Code

Output:

```
mysql  Ver 8.1.0 for FreeBSD14.0 on amd64 (Source distribution)
```

4. Enable the MySQL server to automatically start at boot.

   console Copy

   ```
   $ sudo service mysql enable
   ```

   Explain Code

   Output:

   ```
   mysql enabled in /etc/rc.conf
   ```

5. Start the MySQL database server.

   console Copy

   ```
   $ sudo service mysql-server start
   ```

   Explain Code

6. View the MySQL server status and verify that it's running.

   console Copy

   ```
   $ sudo service mysql-server status
   ```

   Explain Code

   Output:

   ```
   mysql is running as pid 3266.
   ```

7. Start the MySQL secure installation script.

   console Copy

   ```
   $ sudo mysql_secure_installation
   ```

   Explain Code

   - Enter `Y` when prompted to enable the **VALIDATE PASSWORD** component and set strict password policies on the database server.

   ```
   VALIDATE PASSWORD COMPONENT can be used to test passwords
   and improve security. It checks the strength of password
   and allows the users to set only those passwords which are
   secure enough. Would you like to setup VALIDATE PASSWORD component?
   ```

```
Press y|Y for Yes, any other key for No: Y
```

- Enter your desired password strength level to enable on your server. For example, enter 2 to require strong passwords with mixed characters.

```
There are three levels of password validation policy:

LOW    Length >= 8
MEDIUM Length >= 8, numeric, mixed case, and special characters
STRONG Length >= 8, numeric, mixed case, special characters and dictionary        file

Please enter 0 = LOW, 1 = MEDIUM and 2 = STRONG:
```

- Enter a new strong password to assign the `root` database user.

```
Please set the password for root here.

New password:
```

- Repeat the password and verify the estimated password strength.

```
Re-enter new password:

Estimated strength of the password: 8
```

- Enter `Y` when prompted to validate and continue with the new user password.

```
Do you wish to continue with the password provided?(Press y|Y for Yes, any other key for No) :
```

- Enter `Y` and press `Enter` to remove anonymous database users on your server.

```
Remove anonymous users? (Press y|Y for Yes, any other key for No) :
```

- Enter `Y` when prompted to disable remote access to your database server using the root user account.

```
Disallow root login remotely? (Press y|Y for Yes, any other key for No) :
```

- Enter `Y` when prompted to remove the test databases on your server.

```
`Remove test database and access to it? (Press y|Y for Yes, any other key for No) :
```

- Enter `Y` and press `Enter` to reload the MySQL privilege tables and apply your configuration changes.

```
Reload privilege tables now? (Press y|Y for Yes, any other key for No) :
```

8. Restart the MySQL database server to apply changes.

   console Copy

```
$ sudo service mysql-server restart
```

   Explain Code

Install PHP and Configure PHP-FPM

PHP is available in the default FreeBSD repositories and includes the PHP-FPM package that manages connection requests using pool configurations on your server. Follow the steps below to install the latest PHP version on your server.

1. Search all available PHP packages in the default FreeBSD repositories.

   console Copy

```
$ pkg search php | egrep '^php[0-9]+-[0-9]'
```

   Explain Code

   Output:

```
php81-8.1.29               PHP Scripting Language (8.1.X branch)
php82-8.2.18               PHP Scripting Language (8.2.X branch)
php83-8.3.6                PHP Scripting Language (8.3.X branch)
```

2. Install the latest PHP package. For example, PHP version `8.3`.

   console Copy

```
$ sudo apt install php83
```

   Explain Code

3. Install common PHP modules required to enable application functionalities.

   console Copy

```
$ sudo pkg install -y  php83-mysqli php83-curl php83-zip php83-gd php83-xml php83-mbstring
```

   Explain Code

4. View the installed PHP version on your server..

   console Copy

```
$ php -v
```

Explain Code

Output:

```
PHP 8.3.6 (cli) (built: Jun 20 2024 02:08:30) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.3.6, Copyright (c) Zend Technologies
```

5. View the installed PHP-FPM version on your server.

   console Copy

   ```
   $ php-fpm -v
   ```

   Explain Code

   Output:

   ```
   PHP 8.3.6 (fpm-fcgi) (built: Jun 20 2024 02:08:43)
   Copyright (c) The PHP Group
   Zend Engine v4.3.6, Copyright (c) Zend Technologies
   ```

6. Enable PHP-FPM to automatically start at boot.

   console Copy

   ```
   $ sudo service php-fpm enable
   ```

   Explain Code

7. View the PHP-FPM service status and verify that it's running.

   console Copy

   ```
   $ sudo service php-fpm status
   ```

   Explain Code

   Output:

   ```
   php_fpm is running as pid 2558.
   ```

Configure Apache with PHP-FPM

Apache uses the `mod_proxy` and `mod_proxy_fcgi` modules to process FastCGI requests and connect to the PHP-FPM service. Follow the steps below to enable the required Apache modules and forward all PHP application requests to the PHP-FPM service port `9000`.

1. Open the default PHP-FPM pool configuration file using a text editor such as the Easy Editor (`ee`).

   console Copy

   ```
   $ sudo ee /usr/local/etc/php-fpm.d/www.conf
   ```

   Explain Code

   - Find the `user` and `group` directives and verify that PHP-FPM runs with the web server user profile `www`.

   ini Copy

   ```
   user = www
   group = www
   ```

   Explain Code

   - Find the `listen` directive and verify that PHP-FPM listens for connection requests on the localhost port `9000`.

   ini Copy

   ```
   listen = 127.0.0.1:9000
   ```

   Explain Code

   Save and close the file.
2. Run the following command to enable the `mod_proxy` and `mod_proxy_fcgi` Apache modules on your server.

   console Copy

   ```
   $ sudo sed -i '' -e 's/^#LoadModule proxy_module libexec\/apache24\/mod_proxy.so/LoadModule proxy_modu
   ```

   Explain Code

   The above command uncomments the following Apache configuration directives to enable the modules on your server.

   apacheconf Copy

   ```
   LoadModule proxy_module libexec/apache24/mod_proxy.so
   LoadModule proxy_fcgi_module libexec/apache24/mod_proxy_fcgi.so
   ```

   Explain Code
3. Open the main Apache configuration file.

   console Copy

   ```
   $ sudo ee /usr/local/etc/apache24/httpd.conf
   ```

   Explain Code

4. Add the following configurations at the end of the file to forward all PHP file requests to the PHP-FPM port `9000`.

   apacheconf Copy

   ```
   <FilesMatch "\.php$">
       SetHandler "proxy:fcgi://127.0.0.1:9000"
   </FilesMatch>
   ```

   Explain Code

   Save and close the file.
5. Test the Apache configuration for errors.

   console Copy

   ```
   $ sudo apachectl configtest
   ```

   Explain Code

   Your output should look like the one below when the test is successful:

   ```
   Performing sanity check on apache24 configuration:
   AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using
   127.0.0.1. Set the 'ServerName' directive globally to suppress this message
   Syntax OK
   ```

6. Restart PHP-FPM to apply changes.

   console Copy

   ```
   $ sudo service php-fpm restart
   ```

   Explain Code
7. Restart Apache to apply the configuration changes.

   console Copy

   ```
   $ sudo service apache24 restart
   ```

   Explain Code

Test the Installation

1. Log in to the MySQL database server using the `root` user and password you set earlier.

   console Copy

   ```
   $ mysql -u root -p
   ```

Explain Code

2. Create a new sample database `exampledb`.

sql Copy

```
mysql> create database exampledb;
```

Explain Code

3. Create a new database user such as `dbadmin` with a strong password.

sql Copy

```
mysql> create user 'dbadmin'@'localhost' IDENTIFIED BY 'strong@@password25Bb';
```

Explain Code

4. Grant the `dbadmin` user full privileges to the `exampledb` database.

sql Copy

```
mysql> GRANT ALL PRIVILEGES ON exampledb.* TO 'dbadmin'@'localhost';
```

Explain Code

5. Flush the MySQL privileges table to apply changes.

sql Copy

```
mysql> FLUSH PRIVILEGES;
```

Explain Code

6. Switch to the sample database `exampledb`.

sql Copy

```
mysql> use exampledb;
```

Explain Code

7. Create a new sample table `exampletable` with the following columns to store integer and string values.

sql Copy

```
mysql> CREATE TABLE exampletable (id INT AUTO_INCREMENT PRIMARY KEY, messages VARCHAR(255) NOT
```

Explain Code

The above SQL statement creates a new table with the columns: `id` that stores numeric data, and `messages` that stores mixed characters.

8. Insert sample data into the table. For example, `Greetings from Vultr`.

sql Copy

```
mysql> INSERT INTO exampletable (messages) VALUES ('Greetings from Vultr');
```

Explain Code

9. Exit the MySQL database console.

   sql Copy

   ```
   mysql> EXIT;
   ```

   Explain Code

10. Create a new directory to store your web application files. For example, `app.example.com` in the default Apache web root path.

    console Copy

    ```
    $ sudo mkdir -p /usr/local/www/apache24/app.example.com
    ```

    Explain Code

11. Create a new sample PHP application file in the directory. For example, `index.php`.

    console Copy

    ```
    $ sudo ee /usr/local/www/apache24/app.example.com/index.php
    ```

    Explain Code

12. Add the following contents to the file.

    php Copy

    ```
    <?php
    $servername = "localhost";
    $username = "dbadmin";
    $password = "strong@@password25Bb";
    $dbname = "exampledb";

    // Connect to the MySQL database
    $conn = new mysqli($servername, $username, $password, $dbname);

    // Test the MySQL database connection
    if ($conn->connect_error) {
        die("Database Connection Failed." . $conn->connect_error);
    }

    // Fetch a string from exampletable
    $sql = "SELECT messages FROM exampletable";
    $result = $conn->query($sql);

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        echo "<h1 align='center'> <br> <br>" . $row["messages"]. "</h1>";

    } else {
        echo "<h1 align='center'> <br> No Data Found.</h1>";
    }
    ```

```
$conn->close();
?>
```

Explain Code

Save the file and exit the text editor.

The above PHP application connects to the MySQL database server and queries the `exampledb` database to output values in the `exampledata` column within the `messages` table. The application displays a `Greetings from Vultr` heading when successful or `No Data Found` when unsuccessful.

13. Create a new Apache virtual host configuration file. For example, `app.example.com.conf`.

    console Copy

    ```
    $ sudo ee /usr/local/etc/apache24/Includes/app.example.com.conf
    ```

    Explain Code

14. Add the following configurations to the file.

    apacheconf Copy

    ```
    <VirtualHost *:80>
        ServerAdmin webmaster@yourdomain.com
        DocumentRoot "/usr/local/www/apache24/app.example.com/"
        ServerName app.example.com

        <Directory "/usr/local/www/apache24/app.example.com/">
            Options Indexes FollowSymLinks
            AllowOverride All
            Require all granted
            DirectoryIndex index.php index.html

        </Directory>

        ErrorLog "/var/log/app.example.com-error.log"
    </VirtualHost>
    ```

    Explain Code

    Save the file and exit the text editor.

    The above Apache configuration creates a new virtual host that listens for connections using the domain `app.example.com` on the default HTTP port `80`. All PHP requests are forwarded to PHP-FPM on the host port `9000` applied in your main Apache configuration.

15. Test the Apache configuration for errors.

    console Copy

    ```
    $ sudo apachectl configtest
    ```

    Explain Code

Output:

```
Performing sanity check on apache24 configuration:
AH00558: httpd: Could not reliably determine the server's fully qualified domain name, using
127.0.0.1. Set the 'ServerName' directive globally to suppress this message
Syntax OK
```

16. Restart Apache to apply the new virtual host configuration.

    console Copy

    ```
    $ sudo service apache24 restart
    ```

    Explain Code

17. Access your domain using a web browser and verify that you can access the PHP web application.

    ```
    http://app.example.com
    ```

    A Sample PHP web application

    Image not found or type unknown

Secure the Server

1. Install the Certbot Let's Encrypt client for Apache.

   console Copy

   ```
   $ sudo pkg install py39-certbot-apache
   ```

   Explain Code

2. Run the following commands to enable the Apache `mod_rewrite` and `mod_ssl` modules to allow SSL configurations on your web server.

   console Copy

   ```
   $ sudo sed -i '' 's/#LoadModule rewrite_module libexec\/apache24\/mod_rewrite.so/LoadModule rewrite_modu

   $ sudo sed -i '' 's/#LoadModule ssl_module libexec\/apache24\/mod_ssl.so/LoadModule ssl_module libexec\/ap
   ```

   Explain Code

   The above commands uncomment and enable the following directives in the `/usr/local/etc/apache24/httpd.conf` Apache configuration file:

   apacheconf Copy

   ```
   LoadModule ssl_module libexec/apache24/mod_ssl.so
   LoadModule rewrite_module libexec/apache24/mod_rewrite.so
   ```

   Explain Code

3. Generate a new SSL certificate for your domain. Replace `app.example.com` with your actual domain.

   console Copy

   ```
   $ sudo certbot --apache -d app.example.com --agree-tos
   ```

   Explain Code

4. Test the Certbot SSL certificate automatic renewal process.

   console Copy

   ```
   $ sudo certbot renew --dry-run
   ```

   Explain Code

   Output:

   ```
   Account registered.
   Simulating renewal of an existing certificate for app.example.com

   - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
   Congratulations, all simulated renewals succeeded:
     /usr/local/etc/letsencrypt/live/app.example.com/fullchain.pem (success)
   - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
   ```

5. Restart Apache to apply your SSL configuration changes.

   console Copy

   ```
   $ sudo service apache24 restart
   ```

   Explain Code

6. Create a new Packet Filter (`pf`) firewall configuration.

   console Copy

   ```
   $ sudo ee /etc/pf.conf
   ```

   Explain Code

7. Add the following configurations to the file. Replace `vtnet0` with your public server interface name.

   ini Copy

   ```
   ext_interface = "vtnet0"            # The external server interface

   pass quick on lo0 all      # Allow all connections on the loopback interface

   # Filter rules
   ```

```
block in all               # Block incoming unpermitted traffic
pass out all keep state    # Allow outgoing traffic

pass in on $ext_interface proto tcp from any to any port 22 keep state  # Allow SSH
pass in on $ext_interface proto tcp from any to any port 80 keep state  # Allow HTTP
pass in on $ext_interface proto tcp from any to any port 443 keep state # Allow HTTPS
```

Explain Code

Save the file and exit the text editor.

The above firewall configuration enables network connections to the SSH port `22`, HTTP port `80`, and the HTTPS port `443`. In addition, the firewall blocks any other network connections to other ports on your server.

8. Enable the firewall to automatically start at boot.

   console Copy

   ```
   $ sudo sysrc pf_enable="YES"
   ```

   Explain Code

   Output:

   ```
   pf_enable: NO -> YES
   ```

9. Load your firewall configuration.

   console Copy

   ```
   $ sudo pfctl -f /etc/pf.conf
   ```

   Explain Code

10. Start the Packet Filter firewall.

    console Copy

    ```
    $ sudo pfctl -e
    ```

    Explain Code

    Output:

    ```
    pf enabled
    ```

11. View all firewall rules and verify that your SSH, HTTP, and HTTPS ports are available.

    console Copy

    ```
    $ sudo pfctl -sr
    ```

    Explain Code

    Output:

```
pass quick on lo0 all flags S/SA keep state
block drop in all
pass in on vtnet0 proto tcp from any to any port = ssh flags S/SA keep state
pass in on vtnet0 proto tcp from any to any port = http flags S/SA keep state
pass in on vtnet0 proto tcp from any to any port = https flags S/SA keep state
pass out all flags S/SA keep state
```

12. Access your domain using HTTPS to verify that your SSL certificate is valid and encrypted by the Apache web server.

```
https://app.example.com
```

Conclusion

You have successfully installed the Apache, MySQL, and PHP (FAMP) stack on your FreeBSD 14.0 server. All applications in the stack run collectively on the server to deliver dynamic web applications. You can configure the Apache web server and set up MySQL databases to securely deliver web applications or frameworks such as WordPress on your server.

Revision #1
Created 2 May 2026 06:52:20 by buzz
Updated 2 May 2026 06:53:46 by buzz